US011595688B2

(12) **United States Patent**
Li et al.

(10) **Patent No.: US 11,595,688 B2**
(45) **Date of Patent: Feb. 28, 2023**

(54) **DETERMINATION OF SUB-BLOCK TRANSFORM MODE BASED ON CU PARTITIONS**

(71) Applicant: **Tencent America LLC**, Palo Alto, CA (US)

(72) Inventors: **Xiang Li**, Saratoga, CA (US); **Xin Zhao**, Santa Clara, CA (US); **Liang Zhao**, Sunnyvale, CA (US); **Shan Liu**, San Jose, CA (US)

(73) Assignee: **Tencent America LLC**, Palo Alto, CA (US)

(*) Notice: Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **17/717,012**

(22) Filed: **Apr. 8, 2022**

(65) **Prior Publication Data**

US 2022/0232254 A1 Jul. 21, 2022

Related U.S. Application Data

(63) Continuation of application No. 16/999,753, filed on Aug. 21, 2020, now Pat. No. 11,323,743.

(60) Provisional application No. 62/891,839, filed on Aug. 26, 2019.

(51) **Int. Cl.**

H04N 11/02	(2006.01)
H04N 19/60	(2014.01)
H04N 19/176	(2014.01)

(52) **U.S. Cl.**

CPC ***H04N 19/60*** (2014.11); ***H04N 19/176*** (2014.11)

(58) **Field of Classification Search**

CPC H04N 19/60
USPC 375/240.01–240.29
See application file for complete search history.

(56) **References Cited**

U.S. PATENT DOCUMENTS

10,277,896	B2 *	4/2019	Cote	H04N 19/11
2012/0099642	A1 *	4/2012	Sole	H04N 19/122 375/240.03
2017/0280162	A1 *	9/2017	Zhao	H04N 19/103
2018/0020218	A1 *	1/2018	Zhao	H04N 19/61
2019/0306536	A1 *	10/2019	Lim	H04N 19/149
2020/0288130	A1 *	9/2020	Seregin	H04N 19/12

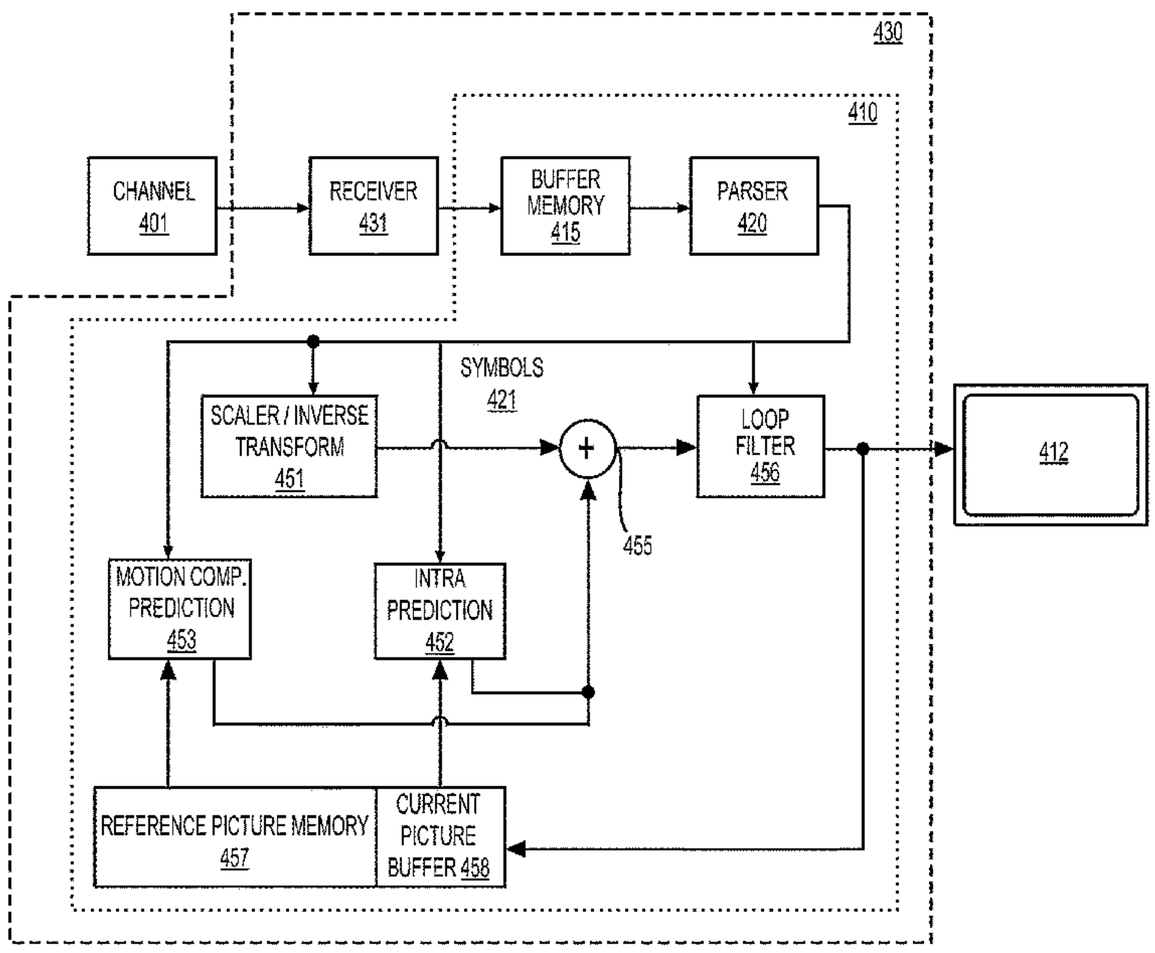
* cited by examiner

Primary Examiner — Leron Beck
(74) *Attorney, Agent, or Firm* — ArentFox Schiff LLP

(57) **ABSTRACT**

Methods, apparatuses, and non-transitory computer-readable storage mediums for video encoding/decoding are provided. In a method, prediction information of a current block of a coding unit tree is generated. The prediction information indicates at least one block partitioning structure is allowed for the current block and a SBT mode is used for the current block. A partition direction and a partition size of the SBT mode for the current block is determined based on the at least one allowed block partitioning structure indicated in the prediction information and based on a comparison between at least one dimension of the current block and a first threshold. A partition of the current block based on the partition direction and the partition size of the SBT mode is different from at least one partition of the current block based on the at least one allowed block partitioning structure indicated in the prediction information.

20 Claims, 19 Drawing Sheets



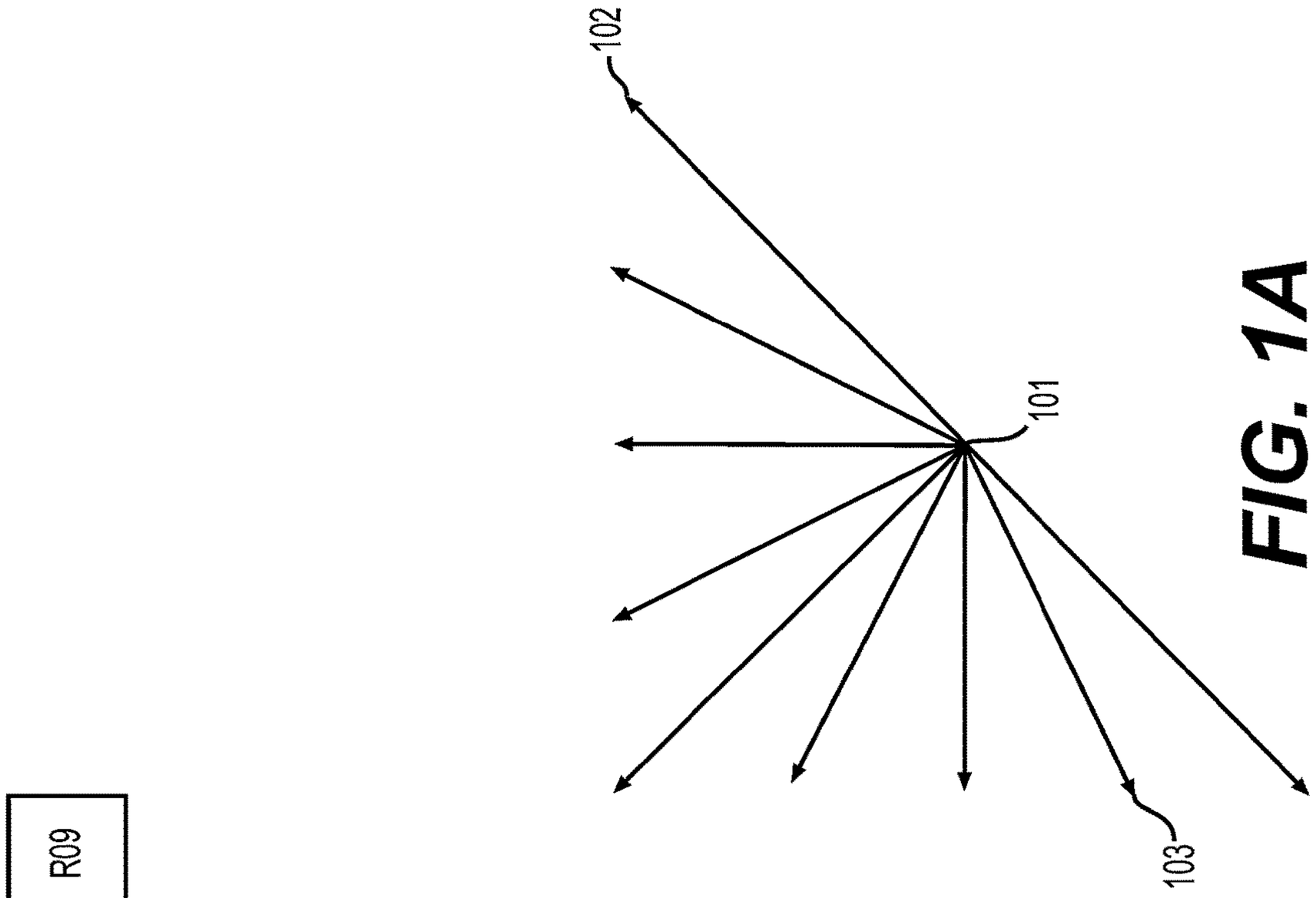


FIG. 1A
(Related Art)

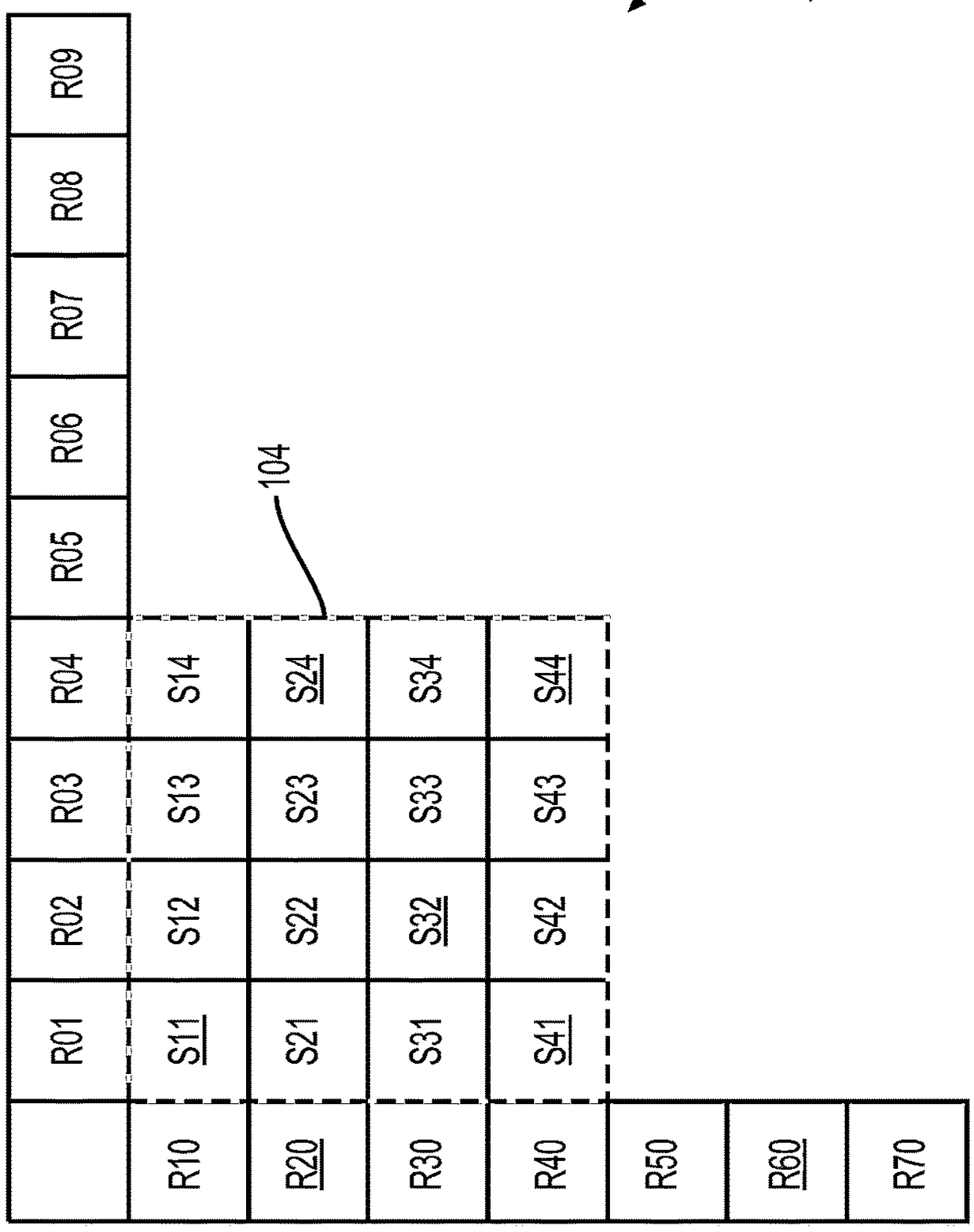
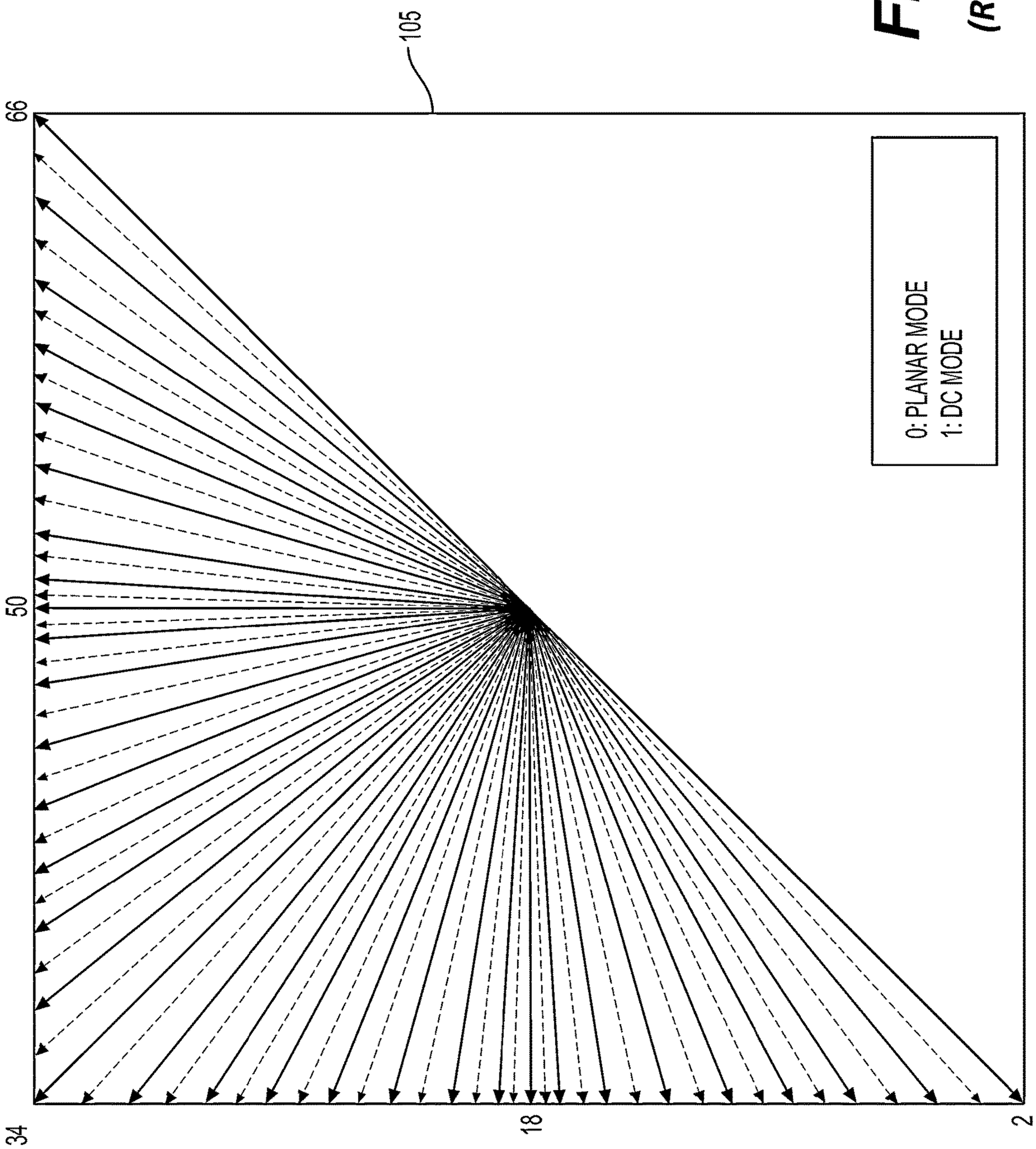


FIG. 1B
(Related Art)



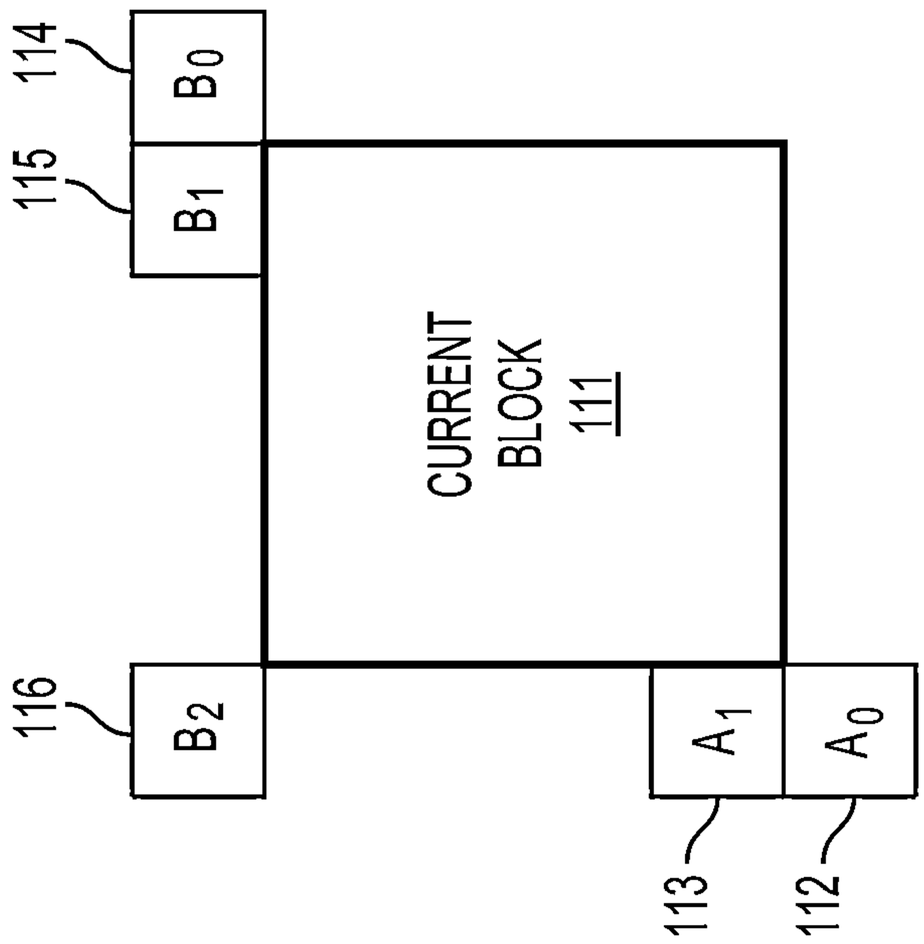


FIG. 1C
(Related Art)

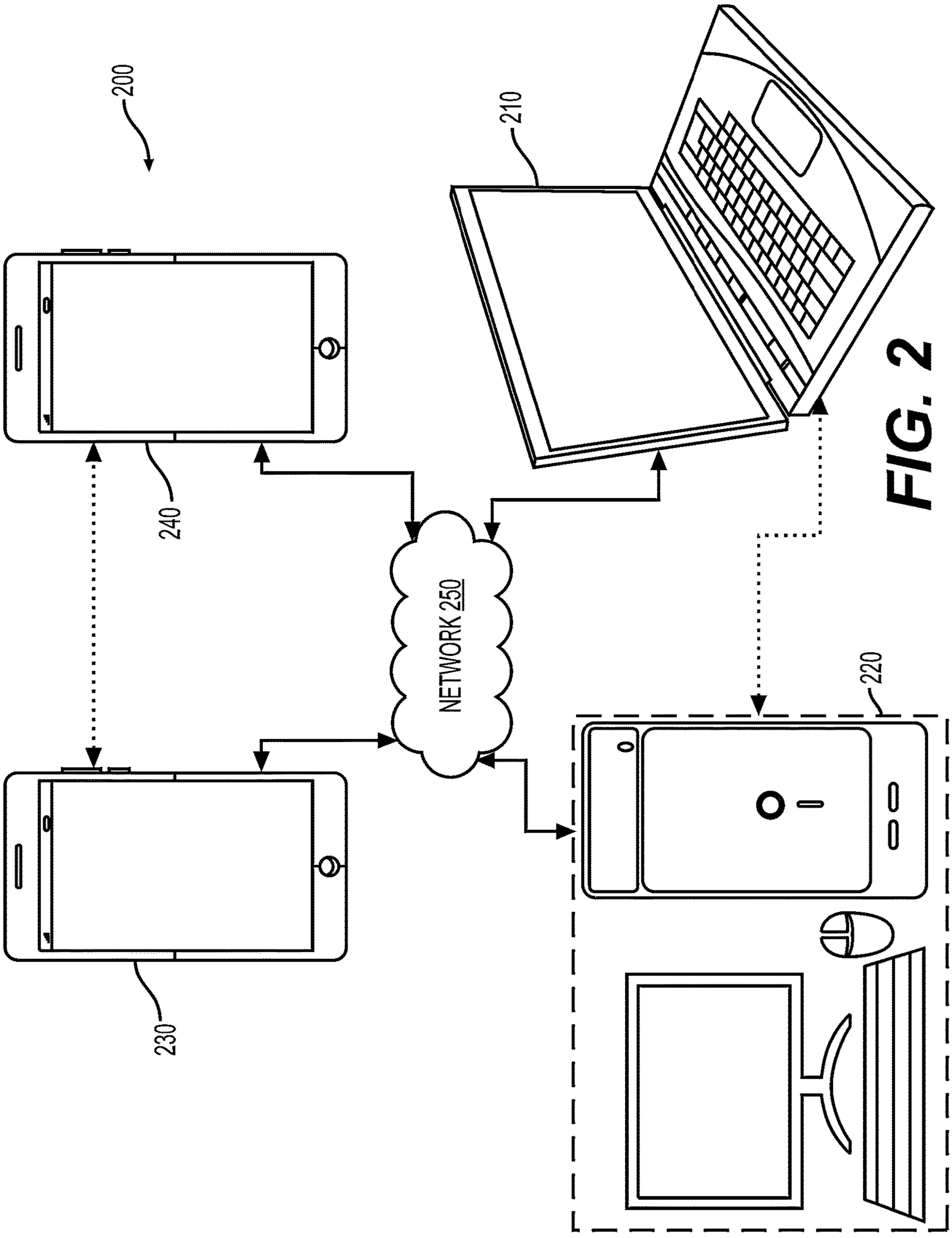


FIG. 2

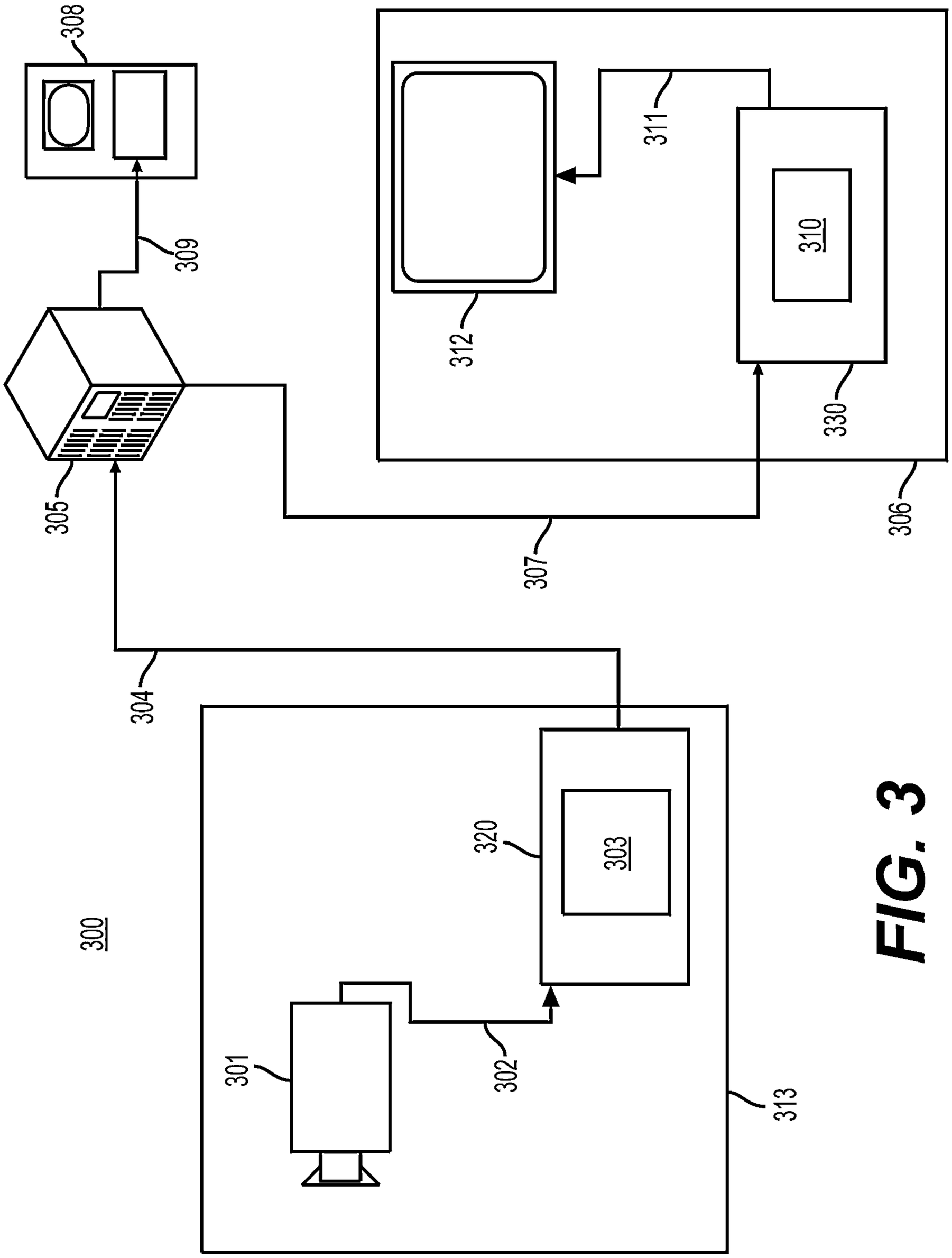
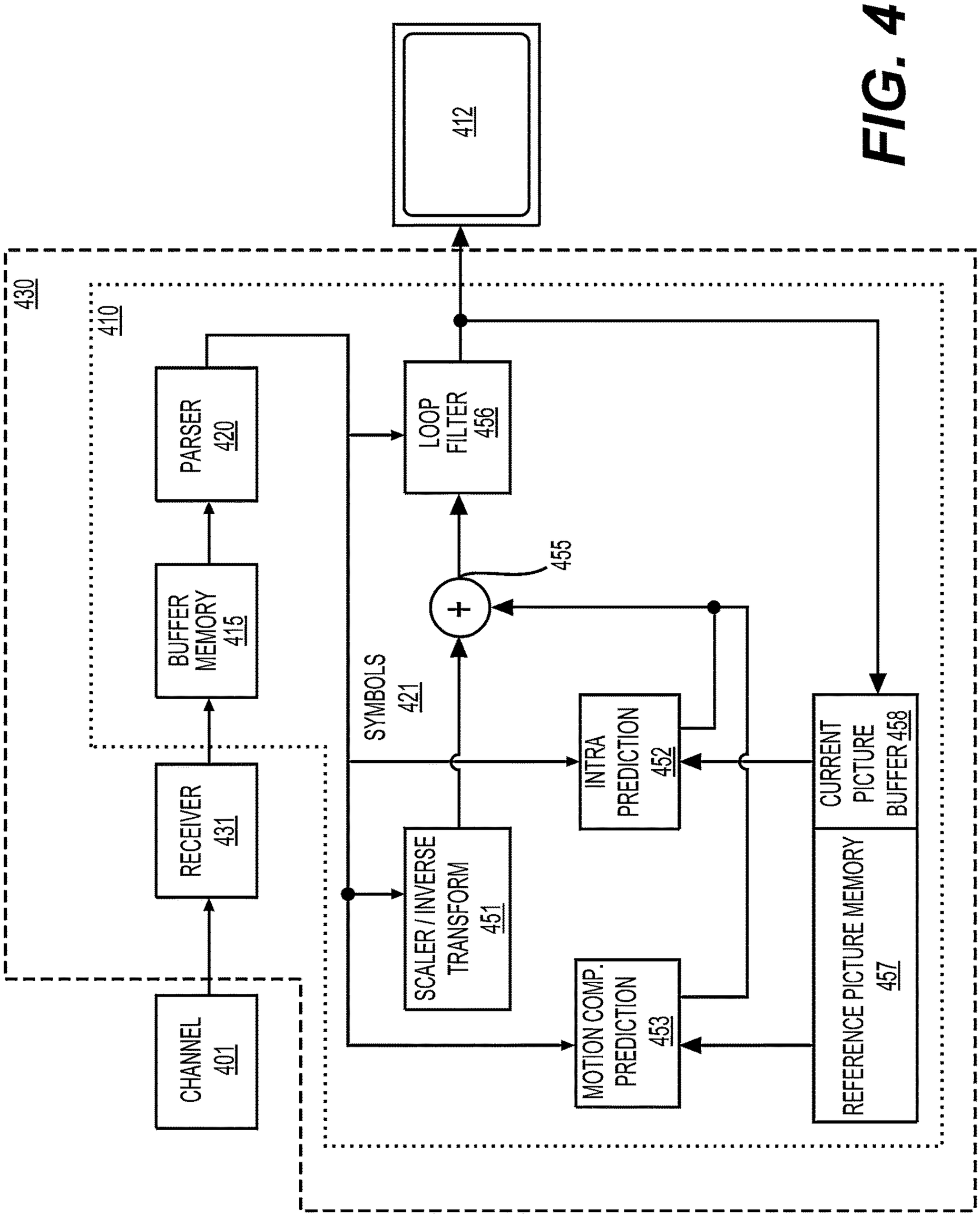


FIG. 3

FIG. 4


412
430
410
PARSER 420
BUFFER MEMORY 415
RECEIVER 431
CHANNEL 401
LOOP FILTER 456
455
+
SYMBOLS 421
INTRA PREDICTION 452
SCALER / INVERSE TRANSFORM 451
MOTION COMP. PREDICTION 453
CURRENT PICTURE BUFFER 458
REFERENCE PICTURE MEMORY 457

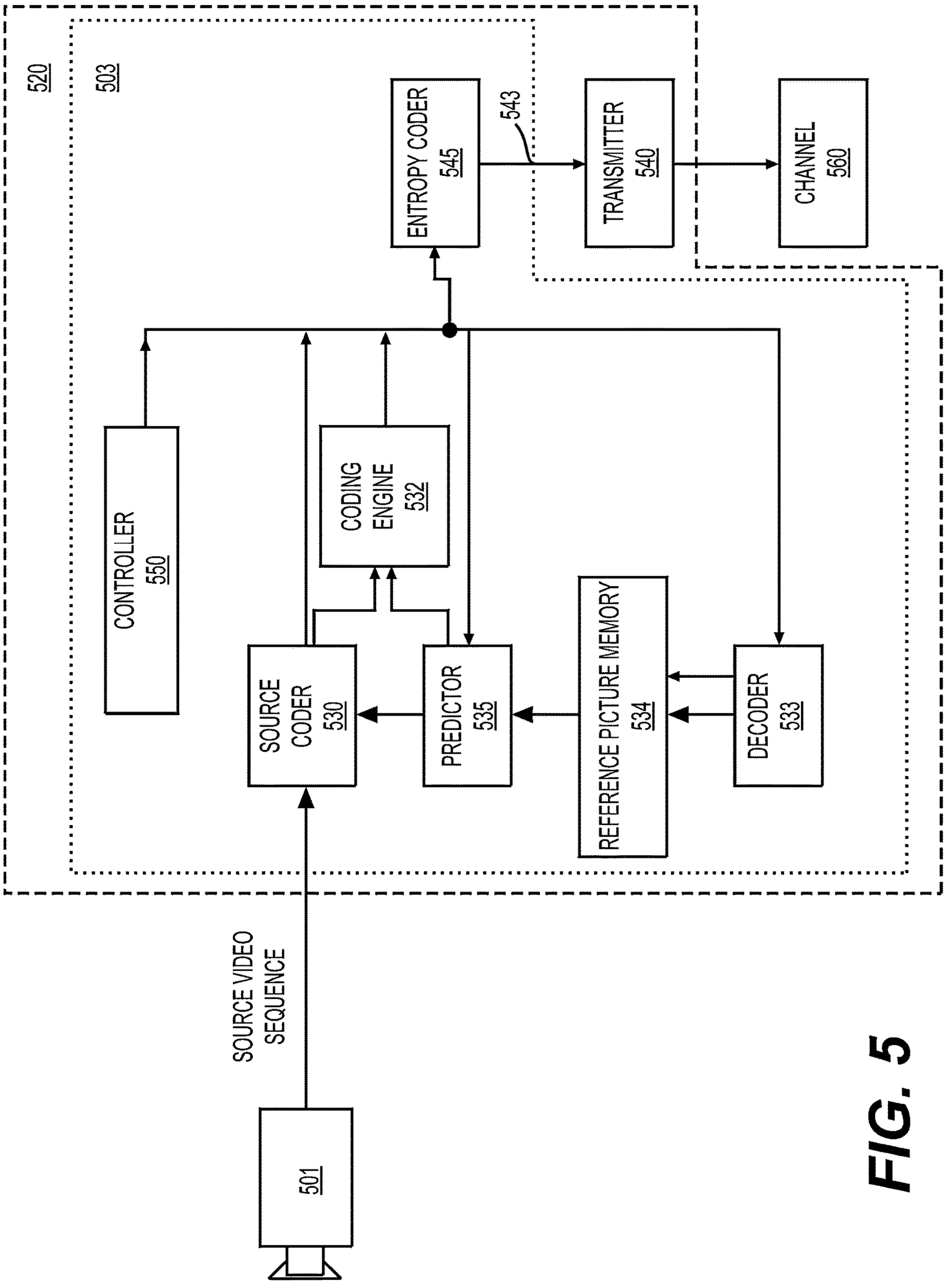

520
503
ENTROPY CODER
545
543
TRANSMITTER
540
CHANNEL
560
CONTROLLER
550
CODING
ENGINE
532
SOURCE
CODER
530
PREDICTOR
535
REFERENCE PICTURE MEMORY
534
DECODER
533
SOURCE VIDEO
SEQUENCE
501

FIG. 5

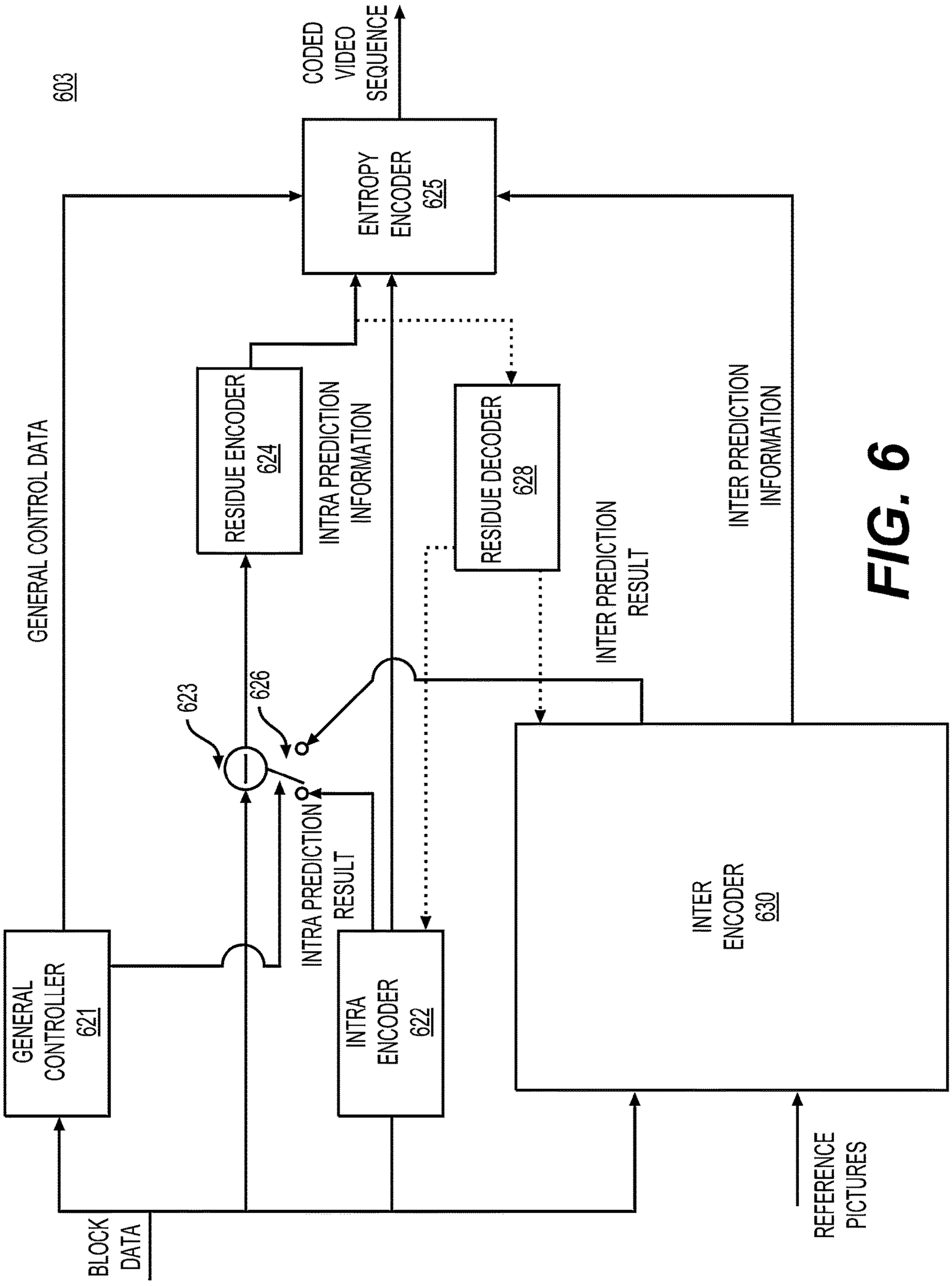

603
CODED VIDEO SEQUENCE
ENTROPY ENCODER 625
RESIDUE ENCODER 624
INTRA PREDICTION INFORMATION
RESIDUE DECODER 628
INTER PREDICTION RESULT
INTER PREDICTION INFORMATION
GENERAL CONTROL DATA
623
626
INTRA PREDICTION RESULT
GENERAL CONTROLLER 621
INTRA ENCODER 622
INTER ENCODER 630
BLOCK DATA
REFERENCE PICTURES

FIG. 6

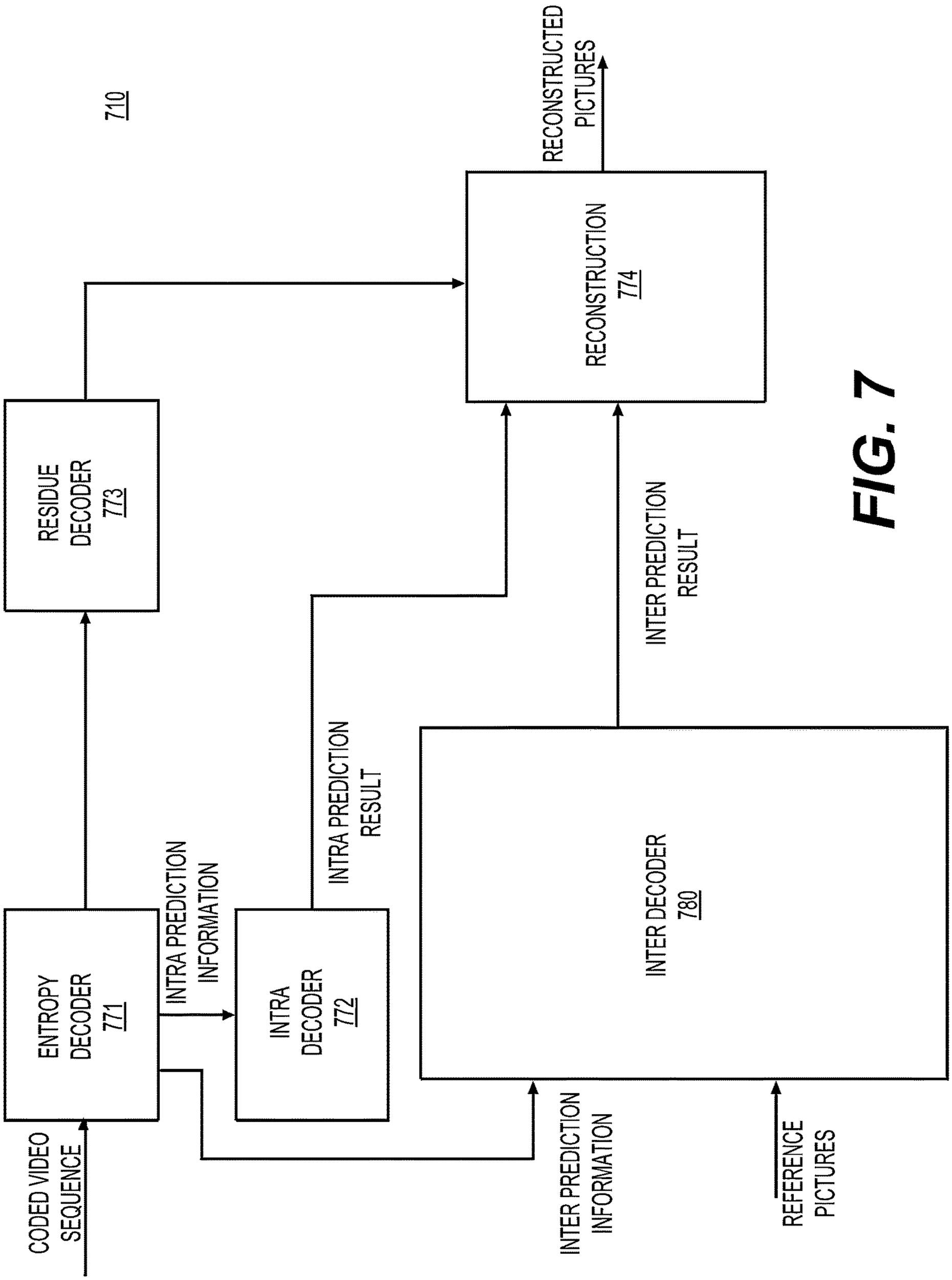

710
RECONSTRUCTED PICTURES
RECONSTRUCTION 774
RESIDUE DECODER 773
INTRA PREDICTION RESULT
INTER PREDICTION RESULT
INTRA PREDICTION INFORMATION
ENTROPY DECODER 771
INTRA DECODER 772
INTER DECODER 780
CODED VIDEO SEQUENCE
INTER PREDICTION INFORMATION
REFERENCE PICTURES

FIG. 7

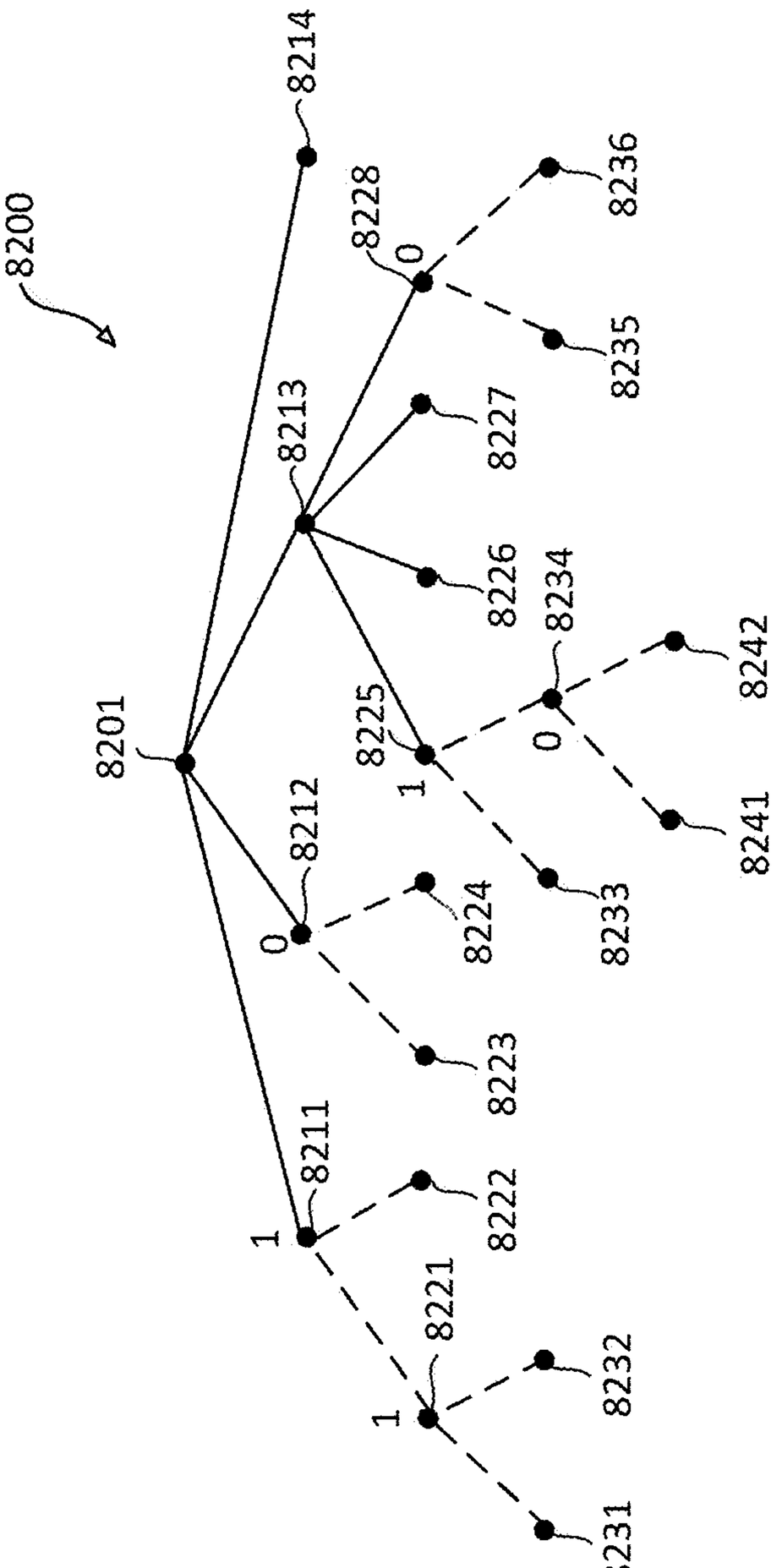

8200
8201
8214
8213
8212
8211
8221
8231
8232
8222
8223
8224
8225
8226
8227
8228
8233
8234
8235
8236
8241
8242
0
1

FIG. 8B

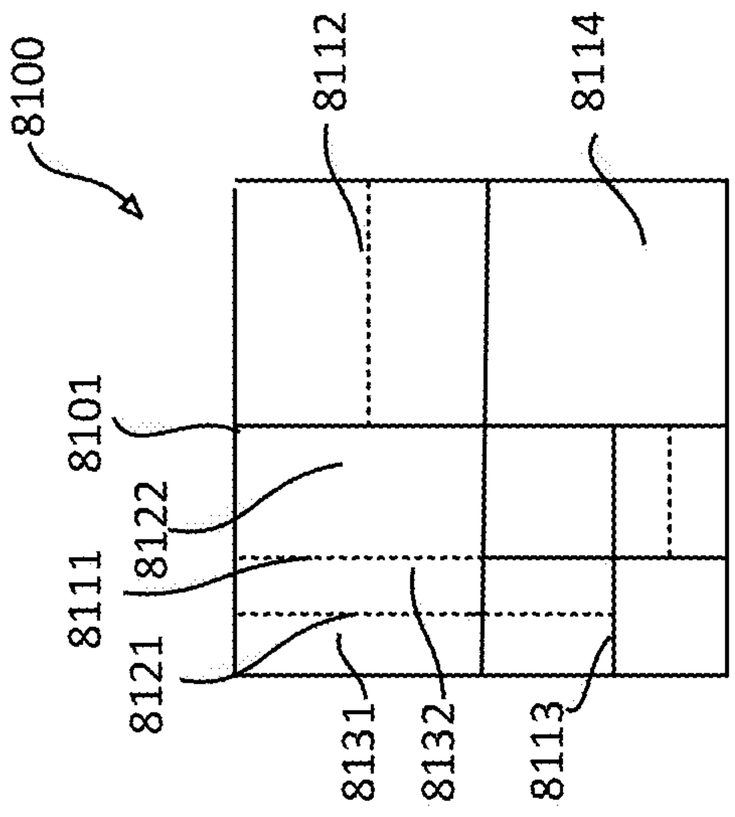

8100
8101
8111
8121
8122
8131
8132
8112
8113
8114

FIG. 8A

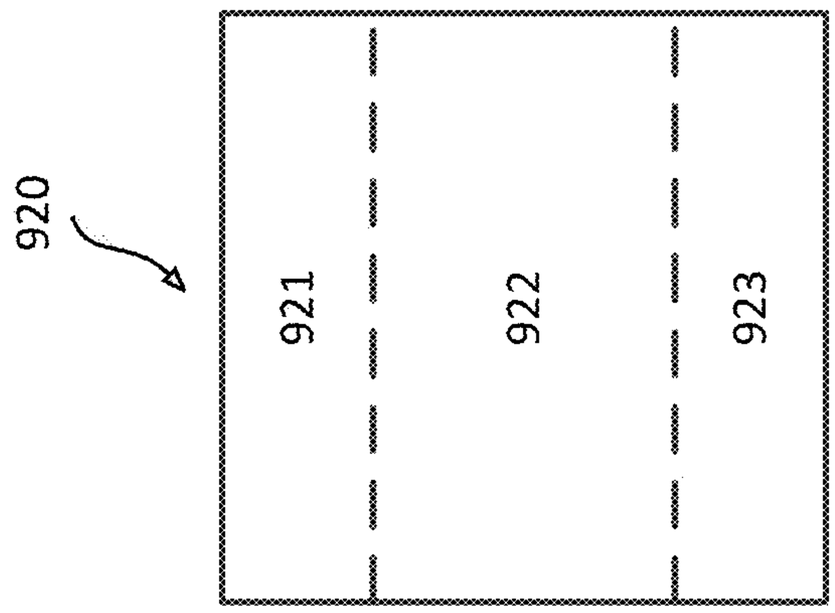


FIG. 9B

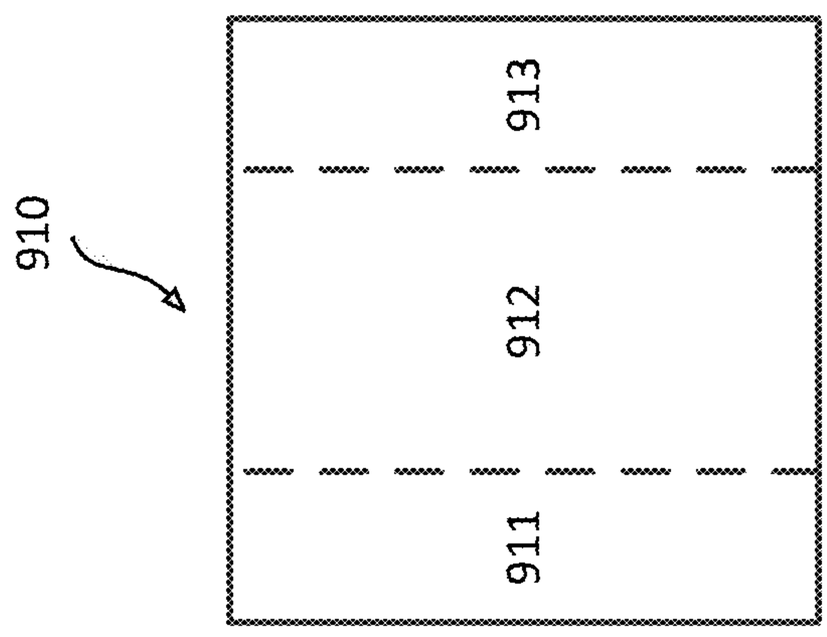


FIG. 9A

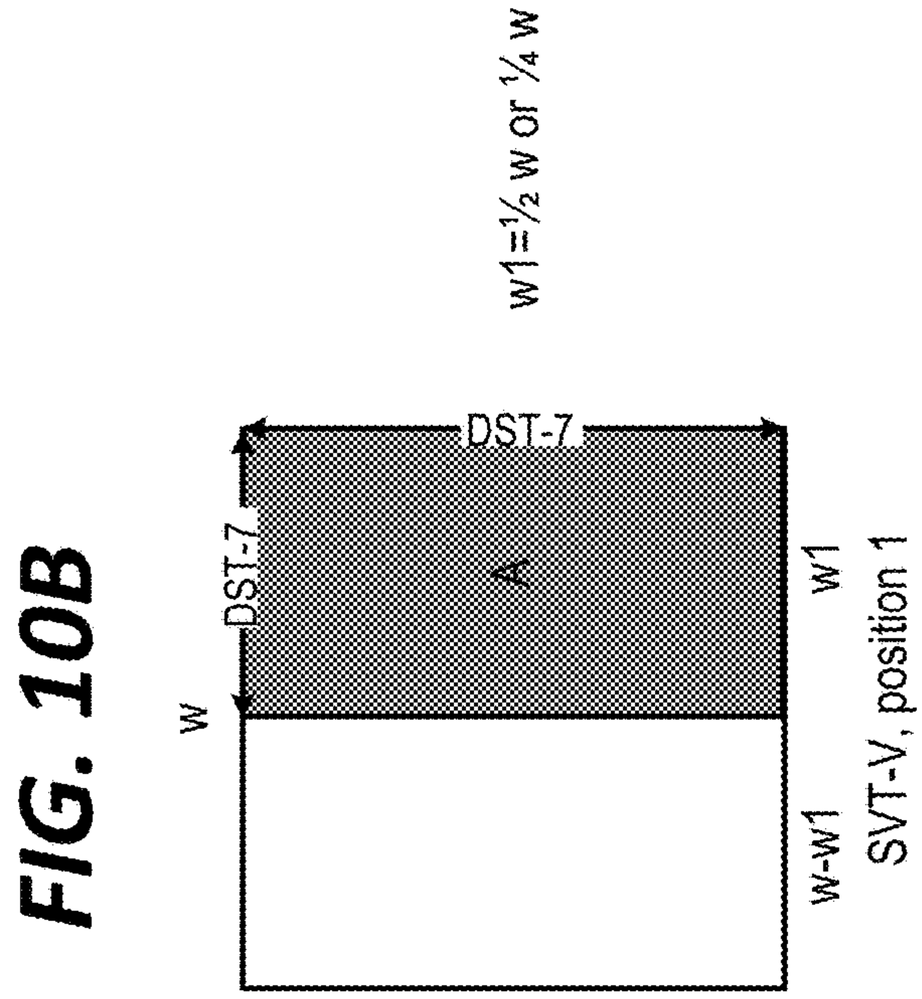


FIG. 10B

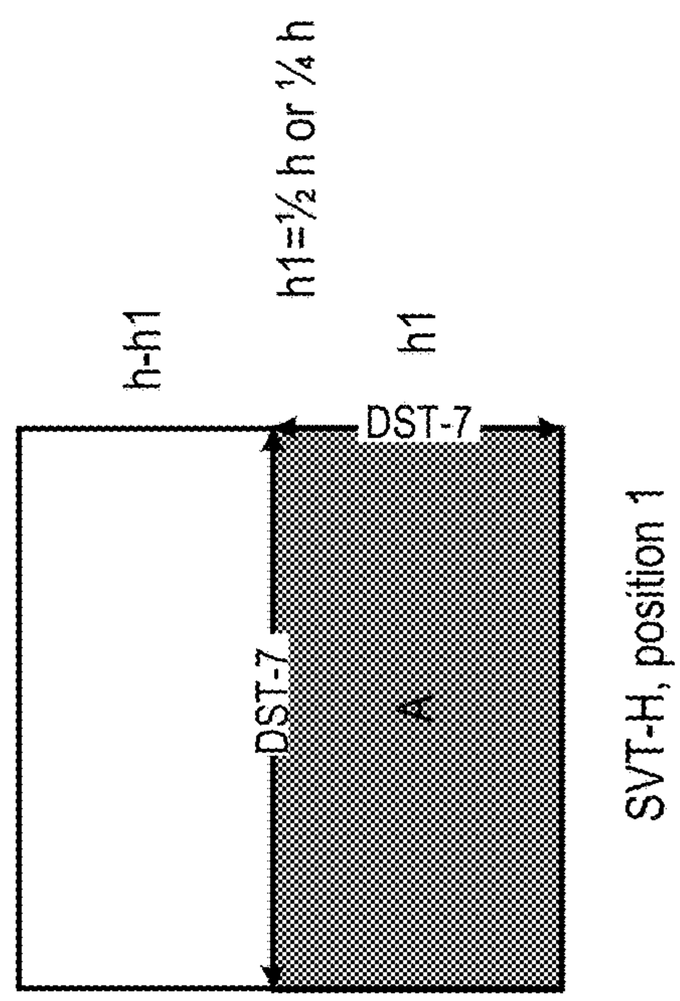


FIG. 10D

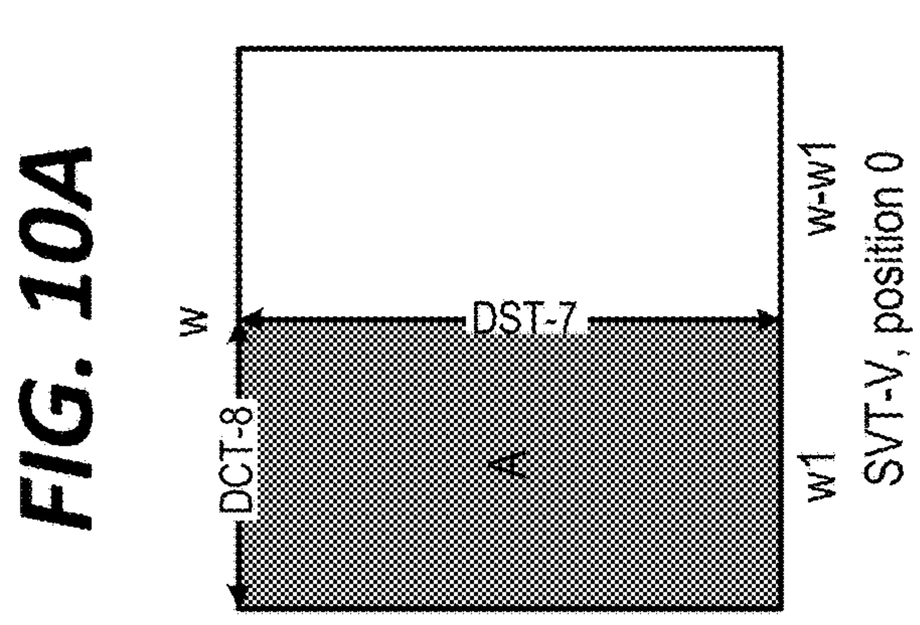


FIG. 10A

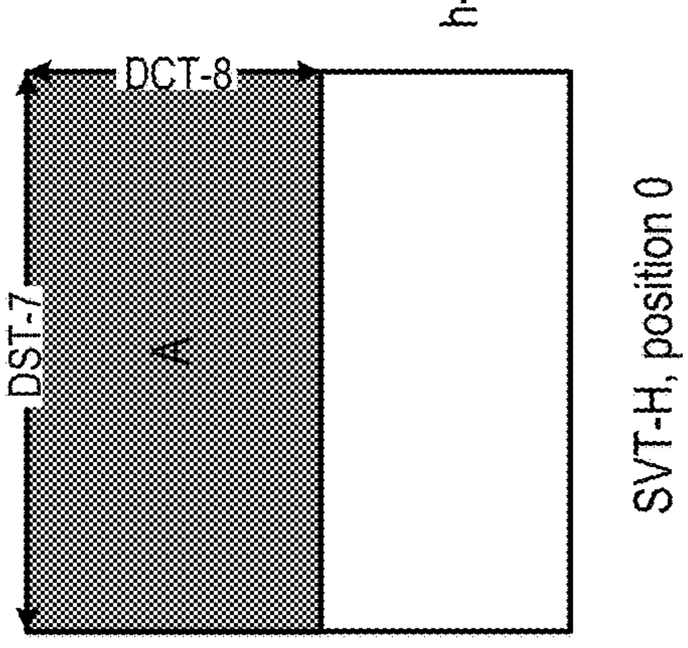


FIG. 10C

Sequence parameter set RBSP syntax

seq_parameter_set_rbsp() {	Descriptor
sps_seq_parameter_set_id	ue(v)
...	
sps_mts_intra_enabled_flag	u(1)
sps_mts_inter_enabled_flag	u(1)
sps_sbt_enable_flag	u(1)
rbsp_trailing_bits()	
}	

FIG. 11

General slice header syntax

slice_header( ) {	Descriptor
slice_pic_parameter_set_id	ue(v)
slice_address	u(v)
slice_type	ue(v)
if (slice_type != I) {	
log2_diff_ctu_max_bt_size	ue(v)
if(sps_sbtmvp_enabled_flag) {	
sbtmvp_size_override_flag	u(1)
if(sbtmvp_size_overide_flag)	
log2_sbtmvp_active_size_minus2	u(3)
}	
if(sps_temporal_mvp_enabled_flag)	
slice_temporal_mvp_enabled_flag	u(1)
if(slice_type == B)	
mvd_ll_zero_flag	u(1)
if(slice_temporal_mvp_enabled_flag) {	
if(slice_type == B)	
collocated_from_l0_flag	u(1)
}	
six_minus_max_num_merge_cand	ue(v)
if(sps_sbt_enable_flag)	
slice_max_sbt_size_64_flag	u(1)
}	
if (sps_alf_enabled_flag) {	
slice_alf_enabled_flag	u(1)
if(slice_alf_enabled_flag)	
alf_data( )	
}	
dep_quant_enabled_flag	u(1)
if(!dep_quant_enabled_flag)	
sign_data_hiding_enabled_flag	u(1)
byte_alignment( )	
}	

FIG. 12

Coding unit syntax

coding_unit(x0, y0, cbWidth, cbHeight, treeType) {	Descriptor
...	
if(CuPredMode[x0][y0] != MODE_INTRA && cu_skip_flag[x0][y0] == 0)	
cu_cbf	ae(v)
if(cu_cbf) {	
if(CuPredMode[x0][y0] != MODE_INTRA && sps_sbt_enable_flag) {	
if(cbWidth <= maxSbtSize && cbHeight <= maxSbtSize) {	
allowSbtVerHalf = cbWidth >= 8	
allowSbtVerQuad = cbWidth >= 16	
allowSbtHorHalf = cbHeight >= 8	
allowSbtHorQuad = cbHeight >= 16	
if(allowSbtVerHalf || allowSbtHorHalf || allowSbtVerQuad || allowSbtHorQuad)	
cu_sbt_flag[x0][y0]	ae(v)
}	
if(cu_sbt_flag[x0][y0]) {	
if((allowSbtVerHalf || allowSbtHorHalf) && (allowSbtVerQuad || allowSbtHorQuad))	
cu_sbt_quad_flag[x0][y0]	ae(v)
if((cu_sbt_quad_flag[x0][y0] && allowSbtVerQuad && allowSbtHorQuad) || (!cu_sbt_quad_flag[x0][y0] && allowSbtVerHalf && allowSbtHorHalf))	
cu_sbt_horizontal_flag[x0][y0]	ae(v)
cu_sbt_pos_flag[x0][y0]	ae(v)
}	
}	
transform_tree(x0, y0, cbWidth, cbHeight, treeType)	
}	
}	

FIG. 13

Transform tree syntax

transform_tree(x0, y0, tbWidth, tbHeight , treeType) {	Descriptor
if(tbWidth > MaxTbSizeY || tbHeight > MaxTbSizeY) {	
trafoWidth = (tbWidth > MaxTbSizeY) ? (tbWidth / 2) : tbWidth	
trafoHeight = (tbHeight > MaxTbSizeY) ? (tbHeight / 2) : tbHeight	
transform_tree(x0, y0, trafoWidth, trafoHeight)	
if(tbWidth > MaxTbSizeY)	
transform_tree(x0 + trafoWidth, y0, trafoWidth, trafoHeight, treeType)	
if(tbHeight > MaxTbSizeY)	
transform_tree(x0, y0 + trafoHeight, trafoWidth, trafoHeight, treeType)	
if(tbWidth > MaxTbSizeY && tbHeight > MaxTbSizeY)	
transform_tree(x0 + trafoWidth, y0 + trafoHeight, trafoWidth, trafoHeight, treeType)	
} else if(cu_sbt_flag[x0][y0])	
factorTb0 = cu_sbt_quad_flag[x0][y0] ? 1 : 2	
factorTb0 = cu_sbt_pos_flag[x0][y0] ? (4 - factorTb0) : factorTb0	
noResiTb0 = cu_sbt_pos_flag[x0][y0] ? 1 : 0	
if(!cu_sbt_horizontal_flag[x0][y0]) {	
trafoWidth = tbWidth * factorTb0 / 4	
transform_tree(x0, y0, trafoWidth, tbHeight, treeType , noResiTb0)	
transform_tree(x0 + trafoWidth, y0, tbWidth - trafoWidth, tbHeight, treeType , !noResiTb0)	
}	
else {	
trafoHeight = tbHeight * factorTb0 / 4	
transform_tree(x0, y0, tbWidth, trafoHeight, treeType , noResiTb0)	
transform_tree(x0, y0 + trafoHeight, tbWidth, tbHeight - trafoHeight, treeType , !noResiTb0)	
}	
} else {	
transform_unit(x0, y0, tbWidth, tbHeight, treeType , 0)	
}	
}	

FIG. 14

Transform unit syntax

transform_unit(x0, y0, tbWidth, tbHeight, treeType , noResi) {	Descriptor
if((treeType = = SINGLE_TREE || treeType = = DUAL_TREE_LUMA) && !noResi)	
tu_cbf_luma[x0][y0]	ae(v)
if((treeType = = SINGLE_TREE || treeType = = DUAL_TREE_CHROMA) && !noResi) {	
tu_cbf_cb[x0][y0]	ae(v)
tu_cbf_cr[x0][y0]	ae(v)
}	
if(((CuPredMode[x0][y0] = = MODE_INTRA) && sps_mts_intra_enabled_flag) || ((CuPredMode[x0][y0] = = MODE_INTER) && sps_mts_inter_enabled_flag)) && tu_cbf_luma[x0][y0] && treeType != DUAL_TREE_CHROMA && (tbWidth <= 32) && (tbHeight <= 32) && !cu_sbt_flag[x0][y0])	
cu_mts_flag[x0][y0]	ae(v)
if(tu_cbf_luma[x0][y0])	
residual_coding(x0, y0, log2(tbWidth), log2(tbHeight), 0)	
if(tu_cbf_cb[x0][y0])	
residual_coding(x0, y0, log2(tbWidth / 2), log2(tbHeight / 2), 1)	
if(tu_cbf_cr[x0][y0])	
residual_coding(x0, y0, log2(tbWidth / 2), log2(tbHeight / 2), 2)	
}	

FIG. 15

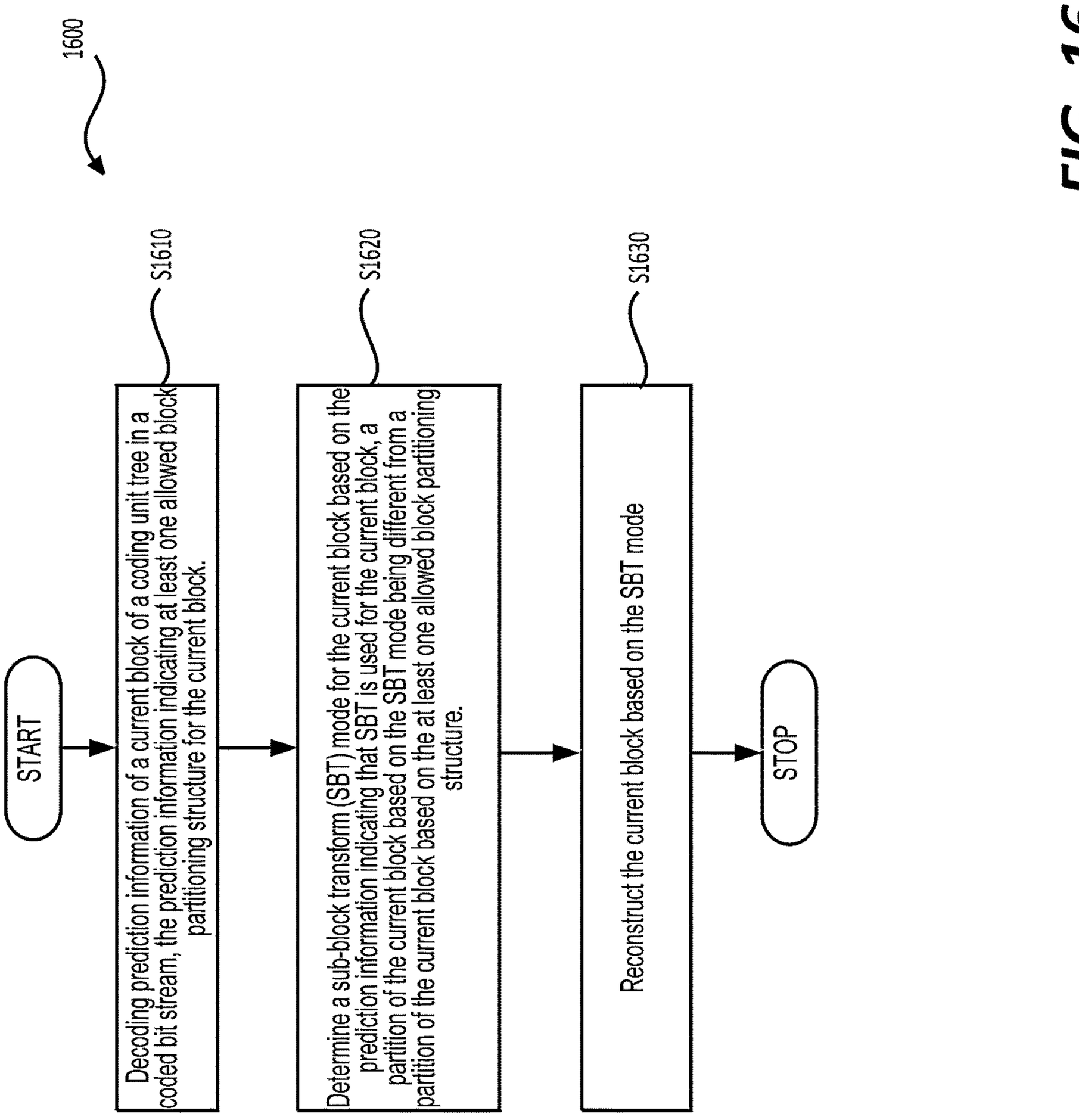

1600
START
Decoding prediction information of a current block of a coding unit tree in a coded bit stream, the prediction information indicating at least one allowed block partitioning structure for the current block.
S1610
Determine a sub-block transform (SBT) mode for the current block based on the prediction information indicating that SBT is used for the current block, a partition of the current block based on the SBT mode being different from a partition of the current block based on the at least one allowed block partitioning structure.
S1620
Reconstruct the current block based on the SBT mode
S1630
STOP

FIG. 16

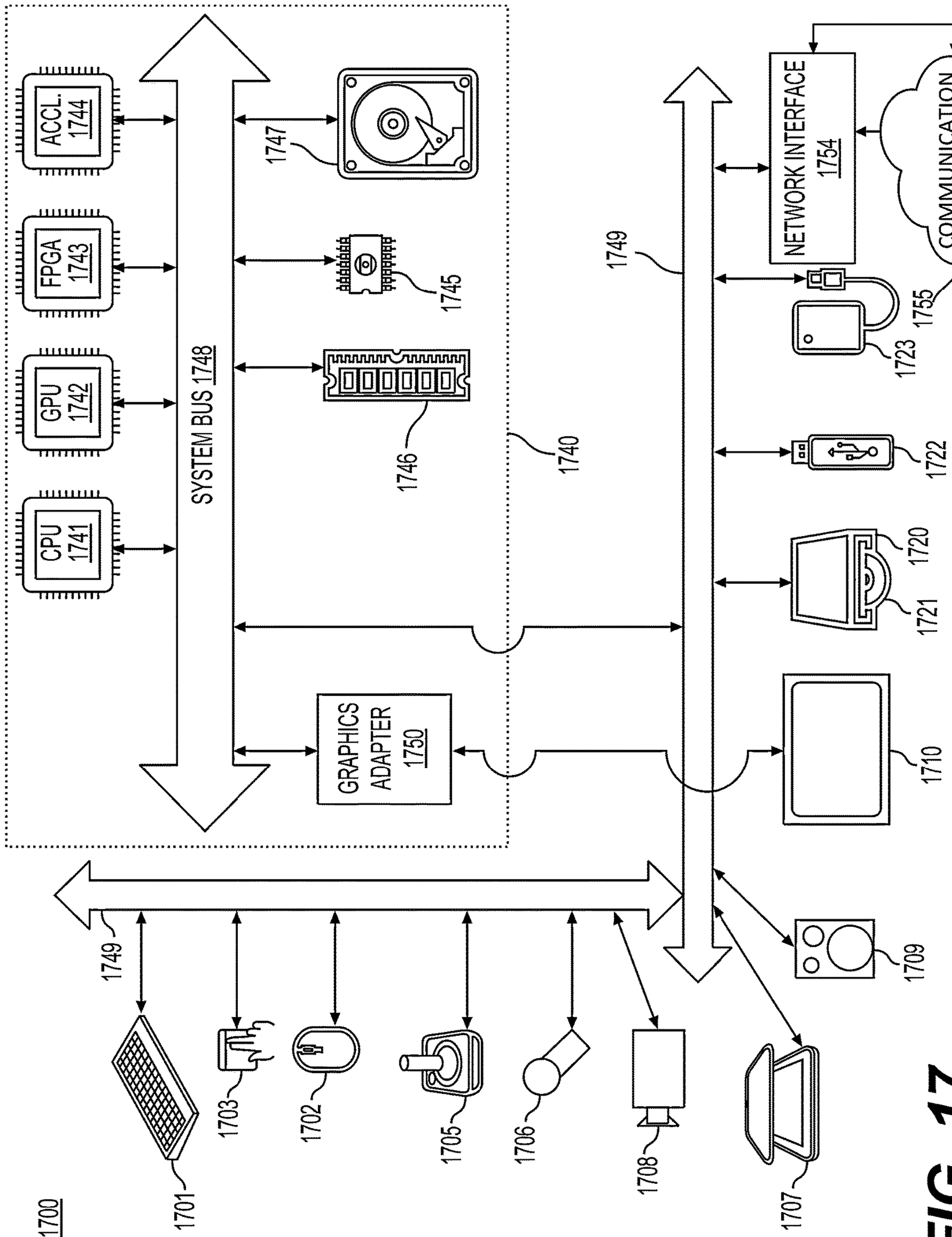

1700
1701
1703
1702
1705
1706
1708
1707
1709
1749
1710
1721
1720
1722
1723
1755
COMMUNICATION NETWORKS
NETWORK INTERFACE 1754
1740
1746
1747
1745
SYSTEM BUS 1748
GRAPHICS ADAPTER 1750
CPU 1741
GPU 1742
FPGA 1743
ACCL. 1744

FIG. 17

DETERMINATION OF SUB-BLOCK TRANSFORM MODE BASED ON CU PARTITIONS

INCORPORATION BY REFERENCE

This present application is a continuation of U.S. Ser. No. 16/999,753, filed on Aug. 21, 2020, which claims the benefit of priority to U.S. Provisional Application No. 62/891,839, "INTERACTION BETWEEN CU PARTITIONS AND SUB-BLOCK TRANSFORM" filed on Aug. 26, 2019. The entire contents of which are incorporated by reference herein in their entirety.

TECHNICAL FIELD

The present disclosure describes embodiments generally related to video coding.

BACKGROUND

The background description provided herein is for the purpose of generally presenting the context of the disclosure. Work of the presently named inventors, to the extent the work is described in this background section, as well as aspects of the description that may not otherwise qualify as prior art at the time of filing, are neither expressly nor impliedly admitted as prior art against the present disclosure.

Video coding and decoding can be performed using inter-picture prediction with motion compensation. Uncompressed digital video can include a series of pictures, each picture having a spatial dimension of, for example, 1920×1080 luminance samples and associated chrominance samples. The series of pictures can have a fixed or variable picture rate (informally also known as frame rate) of, for example, 60 pictures per second or 60 Hz. Uncompressed video has significant bitrate requirements. For example, 1080p60 4:2:0 video at 8 bit per sample (1920×1080 luminance sample resolution at 60 Hz frame rate) requires close to 1.5 Gbit/s bandwidth. An hour of such video requires more than 600 GBytes of storage space.

One purpose of video coding and decoding can be the reduction of redundancy in the input video signal, through compression. Compression can help reduce the aforementioned bandwidth or storage space requirements, in some cases by two orders of magnitude or more. Both lossless and lossy compression, as well as a combination thereof can be employed. Lossless compression refers to techniques where an exact copy of the original signal can be reconstructed from the compressed original signal. When using lossy compression, the reconstructed signal may not be identical to the original signal, but the distortion between original and reconstructed signals is small enough to make the reconstructed signal useful for the intended application. In the case of video, lossy compression is widely employed. The amount of distortion tolerated depends on the application; for example, users of certain consumer streaming applications may tolerate higher distortion than users of television distribution applications. The compression ratio achievable can reflect that: higher allowable/tolerable distortion can yield higher compression ratios.

A video encoder and decoder can utilize techniques from several broad categories, including, for example, motion compensation, transform, quantization, and entropy coding.

Video codec technologies can include techniques known as intra coding. In intra coding, sample values are represented without reference to samples or other data from previously reconstructed reference pictures. In some video codecs, the picture is spatially subdivided into blocks of samples. When all blocks of samples are coded in intra mode, that picture can be an intra picture. Intra pictures and their derivations such as independent decoder refresh pictures, can be used to reset the decoder state and can, therefore, be used as the first picture in a coded video bitstream and a video session, or as a still image. The samples of an intra block can be exposed to a transform, and the transform coefficients can be quantized before entropy coding. Intra prediction can be a technique that minimizes sample values in the pre-transform domain. In some cases, the smaller the DC value after a transform is, and the smaller the AC coefficients are, the fewer the bits that are required at a given quantization step size to represent the block after entropy coding.

Traditional intra coding such as known from, for example MPEG-2 generation coding technologies, does not use intra prediction. However, some newer video compression technologies include techniques that attempt, from, for example, surrounding sample data and/or metadata obtained during the encoding/decoding of spatially neighboring, and preceding in decoding order, blocks of data. Such techniques are henceforth called "intra prediction" techniques. Note that in at least some cases, intra prediction is only using reference data from the current picture under reconstruction and not from reference pictures.

There can be many different forms of intra prediction. When more than one of such techniques can be used in a given video coding technology, the technique in use can be coded in an intra prediction mode. In certain cases, modes can have submodes and/or parameters, and those can be coded individually or included in the mode codeword. Which codeword to use for a given mode/submode/parameter combination can have an impact in the coding efficiency gain through intra prediction, and so can the entropy coding technology used to translate the codewords into a bitstream.

A certain mode of intra prediction was introduced with H.264, refined in H.265, and further refined in newer coding technologies such as joint exploration model (JEM), versatile video coding (VVC), and benchmark set (BMS). A predictor block can be formed using neighboring sample values belonging to already available samples. Sample values of neighboring samples are copied into the predictor block according to a direction. A reference to the direction in use can be coded in the bitstream or may be predicted itself.

Referring to FIG. **1**A, depicted in the lower right is a subset of nine predictor directions known from H.265's 33 possible predictor directions (corresponding to the 33 angular modes of the 35 intra modes). The point where the arrows converge (**101**) represents the sample being predicted. The arrows represent the direction from which the sample is being predicted. For example, arrow (**102**) indicates that sample (**101**) is predicted from a sample or samples to the upper right, at a 45 degree angle from the horizontal. Similarly, arrow (**103**) indicates that sample (**101**) is predicted from a sample or samples to the lower left of sample (**101**), in a 22.5 degree angle from the horizontal.

Still referring to FIG. **1**A, on the top left there is depicted a square block (**104**) of 4×4 samples (indicated by a dashed, boldface line). The square block (**104**) includes 16 samples, each labelled with an "S", its position in the Y dimension (e.g., row index) and its position in the X dimension (e.g., column index). For example, sample S21 is the second sample in the Y dimension (from the top) and the first (from

the left) sample in the X dimension. Similarly, sample S44 is the fourth sample in block (**104**) in both the Y and X dimensions. As the block is 4×4 samples in size, S44 is at the bottom right. Further shown are reference samples that follow a similar numbering scheme. A reference sample is labelled with an R, its Y position (e.g., row index) and X position (column index) relative to block (**104**). In both H.264 and H.265, prediction samples neighbor the block under reconstruction; therefore no negative values need to be used.

Intra picture prediction can work by copying reference sample values from the neighboring samples as appropriated by the signaled prediction direction. For example, assume the coded video bitstream includes signaling that, for this block, indicates a prediction direction consistent with arrow (**102**). That is, samples are predicted from a prediction sample or samples to the upper right, at a 45 degree angle from the horizontal. In that case, samples S41, S32, S23, and S14 are predicted from the same reference sample R05. Sample S44 is then predicted from reference sample R08.

In certain cases, the values of multiple reference samples may be combined, for example through interpolation, in order to calculate a reference sample; especially when the directions are not evenly divisible by 45 degrees.

The number of possible directions has increased as video coding technology has developed. In H.264 (year 2003), nine different direction could be represented. That increased to 33 in H.265 (year 2013), and JEM/VVC/BMS, at the time of disclosure, can support up to 93 directions. Experiments have been conducted to identify the most likely directions, and certain techniques in the entropy coding are used to represent those likely directions in a small number of bits, accepting a certain penalty for less likely directions. Further, the directions themselves can sometimes be predicted from neighboring directions used in neighboring, already decoded, blocks.

FIG. **1**B shows a schematic (**105**) that depicts 65 intra prediction directions according to JEM to illustrate the increasing number of prediction directions over time.

The mapping of intra prediction directions bits in the coded video bitstream that represent the direction can be different from video coding technology to video coding technology; and can range, for example, from simple direct mappings of prediction direction to intra prediction mode, to codewords, to complex adaptive schemes involving most probable modes, and similar techniques. In all cases, however, there can be certain directions that are statistically less likely to occur in video content than certain other directions. As the goal of video compression is the reduction of redundancy, those less likely directions will, in a well working video coding technology, be represented by a larger number of bits than more likely directions.

Motion compensation can be a lossy compression technique and can relate to techniques where a block of sample data from a previously reconstructed picture or part thereof (reference picture), after being spatially shifted in a direction indicated by a motion vector (MV henceforth), is used for the prediction of a newly reconstructed picture or picture part. In some cases, the reference picture can be the same as the picture currently under reconstruction. MVs can have two dimensions X and Y, or three dimensions, the third being an indication of the reference picture in use (the latter, indirectly, can be a time dimension).

In some video compression techniques, an MV applicable to a certain area of sample data can be predicted from other MVs, for example from those related to another area of sample data spatially adjacent to the area under reconstruction, and preceding that MV in decoding order. Doing so can substantially reduce the amount of data required for coding the MV, thereby removing redundancy and increasing compression. MV prediction can work effectively, for example, because when coding an input video signal derived from a camera (known as natural video) there is a statistical likelihood that areas larger than the area to which a single MV is applicable move in a similar direction and, therefore, can in some cases be predicted using a similar motion vector derived from MVs of a neighboring area. That results in the MV found for a given area to be similar or the same as the MV predicted from the surrounding MVs, and that in turn can be represented, after entropy coding, in a smaller number of bits than what would be used if coding the MV directly. In some cases, MV prediction can be an example of lossless compression of a signal (namely: the MVs) derived from the original signal (namely: the sample stream). In other cases, MV prediction itself can be lossy, for example because of rounding errors when calculating a predictor from several surrounding MVs.

Various MV prediction mechanisms are described in H.265/HEVC (ITU-T Rec. H.265, "High Efficiency Video Coding", December 2016). Out of the many MV prediction mechanisms that H.265 offers, described herein is a technique henceforth referred to as "spatial merge."

Referring to FIG. **1**C, a current block (**111**) can include samples that have been found by the encoder during the motion search process to be predictable from a previous block of the same size that has been spatially shifted. Instead of coding that MV directly, the MV can be derived from metadata associated with one or more reference pictures, for example from the most recent (in decoding order) reference picture, using the MV associated with either one of five surrounding samples, denoted A0, A1, and B0, B1, B2 (**112** through **116**, respectively). In H.265, the MV prediction can use predictors from the same reference picture that the neighboring block is using.

SUMMARY

Aspects of the disclosure provide methods and apparatuses for video encoding/decoding. In some examples, an apparatus for video decoding includes processing circuitry.

According to aspects of the disclosure, there is provided a method for video decoding in a decoder. In the method, prediction information of a current block of a coding unit tree in a coded bit stream is decoded. The prediction information indicates at least one allowed block partitioning structure for the current block. A sub-block transform (SBT) mode is determined for the current block based on the prediction information indicating that SBT is used for the current block. A partition of the current block based on the SBT mode is different from a partition of the current block based on the at least one allowed block partitioning structure. The current block is reconstructed based on the SBT mode.

In an embodiment, the SBT mode for the current block is determined when (i) one of a width and a height of the current block is greater than a first threshold and (ii) a partitioning depth associated with the current block is less than a maximum allowed partitioning depth of the coding unit tree. In an example, the SBT mode is determined not to be a vertical SBT mode when the width of the current block is greater than the first threshold. In an example, the SBT mode is determined not to be a horizontal SBT mode when the height of the current block is greater than the first threshold. In an example, the SBT mode is determined not

to be a half SBT mode when (i) the first threshold is 8 and (ii) the at least one allowed partitioning structure includes a binary tree partitioning structure. In an example, the SBT mode is determined not to be a quarter SBT mode when (i) the first threshold is 16 and (ii) the at least one allowed partitioning structure includes a triple tree partitioning structure.

In an embodiment, the SBT mode for the current block is determined when (i) the current block is a chroma block and (ii) sizes of a plurality of sub-blocks of the current block are greater than a second threshold. The current block can be partitioned into the plurality of sub-blocks based on the SBT mode.

In an embodiment, a flag indicating a direction of the SBT mode is not included in the prediction information.

In an embodiment, a flag indicating a size of the SBT mode is not included in the prediction information.

In an embodiment, a flag indicating whether SBT is used for the current block is not included in the prediction information.

In an embodiment, a context model for a flag is determined based on a partitioning depth associated with the current block. The flag indicates whether SBT is used for the current block.

In an embodiment, a context model for a flag is determined based on at least one of a direction of the SBT mode, the width of the current block, and the height of the current block. The flag indicates a size of the SBT mode.

In an embodiment, the prediction information indicates that the current block is a non-merge inter block.

Aspects of the disclosure provide an apparatus configured to perform any one or a combination of the methods for video decoding. In an embodiment, the apparatus includes processing circuitry that decodes prediction information of a current block of a coding unit tree in a coded bit stream is decoded. The prediction information indicates at least one allowed block partitioning structure for the current block. The processing circuitry determines a sub-block transform (SBT) mode for the current block based on the prediction information indicating that SBT is used for the current block. A partition of the current block based on the SBT mode is different from a partition of the current block based on the at least one allowed block partitioning structure. The processing circuitry reconstructs the current block based on the SBT mode.

Aspects of the disclosure also provide a non-transitory computer-readable medium storing instructions which when executed by a computer for video decoding cause the computer to perform any one or a combination of the methods for video decoding.

BRIEF DESCRIPTION OF THE DRAWINGS

Further features, the nature, and various advantages of the disclosed subject matter will be more apparent from the following detailed description and the accompanying drawings in which:

FIG. **1**A shows a schematic illustration of an exemplary subset of intra prediction modes;

FIG. **1**B shows an illustration of exemplary intra prediction directions;

FIG. **1**C shows a schematic illustration of a current block and its surrounding spatial merge candidates in one example;

FIG. **2** shows a schematic illustration of a simplified block diagram of a communication system in accordance with an embodiment;

FIG. **3** shows a schematic illustration of a simplified block diagram of a communication system in accordance with an embodiment;

FIG. **4** shows a schematic illustration of a simplified block diagram of a decoder in accordance with an embodiment;

FIG. **5** shows a schematic illustration of a simplified block diagram of an encoder in accordance with an embodiment;

FIG. **6** shows a block diagram of an encoder in accordance with another embodiment;

FIG. **7** shows a block diagram of a decoder in accordance with another embodiment;

FIGS. **8**A and **8**B illustrate an example of a block partition by using a quad-tree plus binary tree (QTBT) partitioning structure and the corresponding QTBT structure in accordance with an embodiment;

FIGS. **9**A and **9**B show examples of vertical center-side ternary tree partitioning and horizontal center-side ternary tree partitioning, respectively, in accordance with some embodiments;

FIGS. **10**A-**10**D show exemplary sub-block transform (SBT) modes in accordance with some embodiments;

FIGS. **11-15** show exemplary syntax related to SBT methods according to some embodiments of the disclosure;

FIG. **16** shows a flow chart outlining an exemplary process in accordance with an embodiment; and

FIG. **17** shows a schematic illustration of a computer system in accordance with an embodiment.

DETAILED DESCRIPTION OF EMBODIMENTS

The present disclosure includes embodiments directed to unified secondary transform. The embodiments include methods, apparatuses, and non-transitory computer-readable storage mediums for improving secondary transform process. In addition, a block may refer to a prediction block, a coding block, or a coding unit.

I. Video Encoder and Decoder

FIG. **2** illustrates a simplified block diagram of a communication system (**200**) according to an embodiment of the present disclosure. The communication system (**200**) includes a plurality of terminal devices that can communicate with each other, via, for example, a network (**250**). For example, the communication system (**200**) includes a first pair of terminal devices (**210**) and (**220**) interconnected via the network (**250**). In the FIG. **2** example, the first pair of terminal devices (**210**) and (**220**) performs unidirectional transmission of data. For example, the terminal device (**210**) may code video data (e.g., a stream of video pictures that are captured by the terminal device (**210**)) for transmission to the other terminal device (**220**) via the network (**250**). The encoded video data can be transmitted in the form of one or more coded video bitstreams. The terminal device (**220**) may receive the coded video data from the network (**250**), decode the coded video data to recover the video pictures and display video pictures according to the recovered video data. Unidirectional data transmission may be common in media serving applications and the like.

In another example, the communication system (**200**) includes a second pair of terminal devices (**230**) and (**240**) that performs bidirectional transmission of coded video data that may occur, for example, during videoconferencing. For bidirectional transmission of data, in an example, each terminal device of the terminal devices (**230**) and (**240**) may code video data (e.g., a stream of video pictures that are captured by the terminal device) for transmission to the other terminal device of the terminal devices (**230**) and (**240**) via the network (**250**). Each terminal device of the terminal

devices (**230**) and (**240**) also may receive the coded video data transmitted by the other terminal device of the terminal devices (**230**) and (**240**), and may decode the coded video data to recover the video pictures and may display video pictures at an accessible display device according to the recovered video data.

In the FIG. **2** example, the terminal devices (**210**), (**220**), (**230**) and (**240**) may be illustrated as servers, personal computers and smart phones but the principles of the present disclosure may be not so limited. Embodiments of the present disclosure find application with laptop computers, tablet computers, media players and/or dedicated video conferencing equipment. The network (**250**) represents any number of networks that convey coded video data among the terminal devices (**210**), (**220**), (**230**) and (**240**), including for example wireline (wired) and/or wireless communication networks. The communication network (**250**) may exchange data in circuit-switched and/or packet-switched channels. Representative networks include telecommunications networks, local area networks, wide area networks and/or the Internet. For the purposes of the present discussion, the architecture and topology of the network (**250**) may be immaterial to the operation of the present disclosure unless explained herein below.

FIG. **3** illustrates, as an example for an application for the disclosed subject matter, the placement of a video encoder and a video decoder in a streaming environment. The disclosed subject matter can be equally applicable to other video enabled applications, including, for example, video conferencing, digital TV, storing of compressed video on digital media including CD, DVD, memory stick, and the like.

A streaming system may include a capture subsystem (**313**) that can include a video source (**301**), for example a digital camera, creating for example a stream of video pictures (**302**) that are uncompressed. In an example, the stream of video pictures (**302**) includes samples that are taken by the digital camera. The stream of video pictures (**302**), depicted as a bold line to emphasize a high data volume when compared to encoded video data (**304**) (or coded video bitstreams), can be processed by an electronic device (**320**) that includes a video encoder (**303**) coupled to the video source (**301**). The video encoder (**303**) can include hardware, software, or a combination thereof to enable or implement aspects of the disclosed subject matter as described in more detail below. The encoded video data (**304**) (or encoded video bitstream (**304**)), depicted as a thin line to emphasize the lower data volume when compared to the stream of video pictures (**302**), can be stored on a streaming server (**305**) for future use. One or more streaming client subsystems, such as client subsystems (**306**) and (**308**) in FIG. **3** can access the streaming server (**305**) to retrieve copies (**307**) and (**309**) of the encoded video data (**304**). A client subsystem (**306**) can include a video decoder (**310**), for example, in an electronic device (**330**). The video decoder (**310**) decodes the incoming copy (**307**) of the encoded video data and creates an outgoing stream of video pictures (**311**) that can be rendered on a display (**312**) (e.g., display screen) or other rendering device (not depicted). In some streaming systems, the encoded video data (**304**), (**307**), and (**309**) (e.g., video bitstreams) can be encoded according to certain video coding/compression standards. Examples of those standards include ITU-T Recommendation H.265. In an example, a video coding standard under development is informally known as Versatile Video Coding (VVC). The disclosed subject matter may be used in the context of VVC.

It is noted that the electronic devices (**320**) and (**330**) can include other components (not shown). For example, the electronic device (**320**) can include a video decoder (not shown) and the electronic device (**330**) can include a video encoder (not shown) as well.

FIG. **4** shows a block diagram of a video decoder (**410**) according to an embodiment of the present disclosure. The video decoder (**410**) can be included in an electronic device (**430**). The electronic device (**430**) can include a receiver (**431**) (e.g., receiving circuitry). The video decoder (**410**) can be used in the place of the video decoder (**310**) in the FIG. **3** example.

The receiver (**431**) may receive one or more coded video sequences to be decoded by the video decoder (**410**); in the same or another embodiment, one coded video sequence at a time, where the decoding of each coded video sequence is independent from other coded video sequences. The coded video sequence may be received from a channel (**401**), which may be a hardware/software link to a storage device which stores the encoded video data. The receiver (**431**) may receive the encoded video data with other data, for example, coded audio data and/or ancillary data streams, that may be forwarded to their respective using entities (not depicted). The receiver (**431**) may separate the coded video sequence from the other data. To combat network jitter, a buffer memory (**415**) may be coupled in between the receiver (**431**) and an entropy decoder/parser (**420**) ("parser (**420**)" henceforth). In certain applications, the buffer memory (**415**) is part of the video decoder (**410**). In others, it can be outside of the video decoder (**410**) (not depicted). In still others, there can be a buffer memory (not depicted) outside of the video decoder (**410**), for example to combat network jitter, and in addition another buffer memory (**415**) inside the video decoder (**410**), for example to handle playout timing. When the receiver (**431**) is receiving data from a store/forward device of sufficient bandwidth and controllability, or from an isosynchronous network, the buffer memory (**415**) may not be needed, or can be small. For use on best effort packet networks such as the Internet, the buffer memory (**415**) may be required, can be comparatively large and can be advantageously of adaptive size, and may at least partially be implemented in an operating system or similar elements (not depicted) outside of the video decoder (**410**).

The video decoder (**410**) may include the parser (**420**) to reconstruct symbols (**421**) from the coded video sequence. Categories of those symbols include information used to manage operation of the video decoder (**410**), and potentially information to control a rendering device such as a render device (**412**) (e.g., a display screen) that is not an integral part of the electronic device (**430**) but can be coupled to the electronic device (**430**), as was shown in FIG. **4**. The control information for the rendering device(s) may be in the form of Supplemental Enhancement Information (SEI messages) or Video Usability Information (VUI) parameter set fragments (not depicted). The parser (**420**) may parse/entropy-decode the coded video sequence that is received. The coding of the coded video sequence can be in accordance with a video coding technology or standard, and can follow various principles, including variable length coding, Huffman coding, arithmetic coding with or without context sensitivity, and so forth. The parser (**420**) may extract from the coded video sequence, a set of subgroup parameters for at least one of the subgroups of pixels in the video decoder, based upon at least one parameter corresponding to the group. Subgroups can include Groups of Pictures (GOPs), pictures, tiles, slices, macroblocks, Coding Units (CUs), blocks, Transform Units (TUs), Prediction

Units (PUs) and so forth. The parser (**420**) may also extract from the coded video sequence information such as transform coefficients, quantizer parameter values, motion vectors, and so forth.

The parser (**420**) may perform an entropy decoding/parsing operation on the video sequence received from the buffer memory (**415**), so as to create symbols (**421**).

Reconstruction of the symbols (**421**) can involve multiple different units depending on the type of the coded video picture or parts thereof (such as: inter and intra picture, inter and intra block), and other factors. Which units are involved, and how, can be controlled by the subgroup control information that was parsed from the coded video sequence by the parser (**420**). The flow of such subgroup control information between the parser (**420**) and the multiple units below is not depicted for clarity.

Beyond the functional blocks already mentioned, the video decoder (**410**) can be conceptually subdivided into a number of functional units as described below. In a practical implementation operating under commercial constraints, many of these units interact closely with each other and can, at least partly, be integrated into each other. However, for the purpose of describing the disclosed subject matter, the conceptual subdivision into the functional units below is appropriate.

A first unit is the scaler/inverse transform unit (**451**). The scaler/inverse transform unit (**451**) receives a quantized transform coefficient as well as control information, including which transform to use, block size, quantization factor, quantization scaling matrices, etc. as symbol(s) (**421**) from the parser (**420**). The scaler/inverse transform unit (**451**) can output blocks comprising sample values that can be input into aggregator (**455**).

In some cases, the output samples of the scaler/inverse transform (**451**) can pertain to an intra coded block; that is: a block that is not using predictive information from previously reconstructed pictures, but can use predictive information from previously reconstructed parts of the current picture. Such predictive information can be provided by an intra picture prediction unit (**452**). In some cases, the intra picture prediction unit (**452**) generates a block of the same size and shape of the block under reconstruction, using surrounding already reconstructed information fetched from the current picture buffer (**458**). The current picture buffer (**458**) buffers, for example, partly reconstructed current picture and/or fully reconstructed current picture. The aggregator (**455**), in some cases, adds, on a per sample basis, the prediction information that the intra prediction unit (**452**) has generated to the output sample information as provided by the scaler/inverse transform unit (**451**).

In other cases, the output samples of the scaler/inverse transform unit (**451**) can pertain to an inter coded, and potentially motion compensated block. In such a case, a motion compensation prediction unit (**453**) can access reference picture memory (**457**) to fetch samples used for prediction. After motion compensating the fetched samples in accordance with the symbols (**421**) pertaining to the block, these samples can be added by the aggregator (**455**) to the output of the scaler/inverse transform unit (**451**) (in this case called the residual samples or residual signal) so as to generate output sample information. The addresses within the reference picture memory (**457**) from where the motion compensation prediction unit (**453**) fetches prediction samples can be controlled by motion vectors, available to the motion compensation prediction unit (**453**) in the form of symbols (**421**) that can have, for example X, Y, and reference picture components. Motion compensation also can include interpolation of sample values as fetched from the reference picture memory (**457**) when sub-sample exact motion vectors are in use, motion vector prediction mechanisms, and so forth.

The output samples of the aggregator (**455**) can be subject to various loop filtering techniques in the loop filter unit (**456**). Video compression technologies can include in-loop filter technologies that are controlled by parameters included in the coded video sequence (also referred to as coded video bitstream) and made available to the loop filter unit (**456**) as symbols (**421**) from the parser (**420**), but can also be responsive to meta-information obtained during the decoding of previous (in decoding order) parts of the coded picture or coded video sequence, as well as responsive to previously reconstructed and loop-filtered sample values.

The output of the loop filter unit (**456**) can be a sample stream that can be output to the render device (**412**) as well as stored in the reference picture memory (**457**) for use in future inter-picture prediction.

Certain coded pictures, once fully reconstructed, can be used as reference pictures for future prediction. For example, once a coded picture corresponding to a current picture is fully reconstructed and the coded picture has been identified as a reference picture (by, for example, the parser (**420**)), the current picture buffer (**458**) can become a part of the reference picture memory (**457**), and a fresh current picture buffer can be reallocated before commencing the reconstruction of the following coded picture.

The video decoder (**410**) may perform decoding operations according to a predetermined video compression technology in a standard, such as ITU-T Rec. H.265. The coded video sequence may conform to a syntax specified by the video compression technology or standard being used, in the sense that the coded video sequence adheres to both the syntax of the video compression technology or standard and the profiles as documented in the video compression technology or standard. Specifically, a profile can select certain tools as the only tools available for use under that profile from all the tools available in the video compression technology or standard. Also necessary for compliance can be that the complexity of the coded video sequence is within bounds as defined by the level of the video compression technology or standard. In some cases, levels restrict the maximum picture size, maximum frame rate, maximum reconstruction sample rate (measured in, for example mega-samples per second), maximum reference picture size, and so on. Limits set by levels can, in some cases, be further restricted through Hypothetical Reference Decoder (HRD) specifications and metadata for HRD buffer management signaled in the coded video sequence.

In an embodiment, the receiver (**431**) may receive additional (redundant) data with the encoded video. The additional data may be included as part of the coded video sequence(s). The additional data may be used by the video decoder (**410**) to properly decode the data and/or to more accurately reconstruct the original video data. Additional data can be in the form of, for example, temporal, spatial, or signal noise ratio (SNR) enhancement layers, redundant slices, redundant pictures, forward error correction codes, and so on.

FIG. **5** shows a block diagram of a video encoder (**503**) according to an embodiment of the present disclosure. The video encoder (**503**) is included in an electronic device (**520**). The electronic device (**520**) includes a transmitter (**540**) (e.g., transmitting circuitry). The video encoder (**503**) can be used in the place of the video encoder (**303**) in the FIG. **3** example.

The video encoder (**503**) may receive video samples from a video source (**501**) (that is not part of the electronic device (**520**) in the FIG. **5** example) that may capture video image(s) to be coded by the video encoder (**503**). In another example, the video source (**501**) is a part of the electronic device (**520**).

The video source (**501**) may provide the source video sequence to be coded by the video encoder (**503**) in the form of a digital video sample stream that can be of any suitable bit depth (for example: 8 bit, 10 bit, 12 bit, . . .), any colorspace (for example, BT.601 Y CrCB, RGB, . . .), and any suitable sampling structure (for example Y CrCb 4:2:0, Y CrCb 4:4:4). In a media serving system, the video source (**501**) may be a storage device storing previously prepared video. In a videoconferencing system, the video source (**501**) may be a camera that captures local image information as a video sequence. Video data may be provided as a plurality of individual pictures that impart motion when viewed in sequence. The pictures themselves may be organized as a spatial array of pixels, wherein each pixel can comprise one or more samples depending on the sampling structure, color space, etc. in use. A person skilled in the art can readily understand the relationship between pixels and samples. The description below focuses on samples.

According to an embodiment, the video encoder (**503**) may code and compress the pictures of the source video sequence into a coded video sequence (**543**) in real time or under any other time constraints as required by the application. Enforcing appropriate coding speed is one function of a controller (**550**). In some embodiments, the controller (**550**) controls other functional units as described below and is functionally coupled to the other functional units. The coupling is not depicted for clarity. Parameters set by the controller (**550**) can include rate control related parameters (picture skip, quantizer, lambda value of rate-distortion optimization techniques, . . .), picture size, group of pictures (GOP) layout, maximum motion vector allowed reference area, and so forth. The controller (**550**) can be configured to have other suitable functions that pertain to the video encoder (**503**) optimized for a certain system design.

In some embodiments, the video encoder (**503**) is configured to operate in a coding loop. As an oversimplified description, in an example, the coding loop can include a source coder (**530**) (e.g., responsible for creating symbols, such as a symbol stream, based on an input picture to be coded, and a reference picture(s)), and a (local) decoder (**533**) embedded in the video encoder (**503**). The decoder (**533**) reconstructs the symbols to create the sample data in a similar manner as a (remote) decoder also would create (as any compression between symbols and coded video bitstream is lossless in the video compression technologies considered in the disclosed subject matter). The reconstructed sample stream (sample data) is input to the reference picture memory (**534**). As the decoding of a symbol stream leads to bit-exact results independent of decoder location (local or remote), the content in the reference picture memory (**534**) is also bit exact between the local encoder and remote encoder. In other words, the prediction part of an encoder "sees" as reference picture samples exactly the same sample values as a decoder would "see" when using prediction during decoding. This fundamental principle of reference picture synchronicity (and resulting drift, if synchronicity cannot be maintained, for example because of channel errors) is used in some related arts as well.

The operation of the "local" decoder (**533**) can be the same as of a "remote" decoder, such as the video decoder (**410**), which has already been described in detail above in conjunction with FIG. **4**. Briefly referring also to FIG. **4**, however, as symbols are available and encoding/decoding of symbols to a coded video sequence by an entropy coder (**545**) and the parser (**420**) can be lossless, the entropy decoding parts of the video decoder (**410**), including the buffer memory (**415**) and the parser (**420**) may not be fully implemented in the local decoder (**533**).

An observation that can be made at this point is that any decoder technology except the parsing/entropy decoding that is present in a decoder also necessarily needs to be present, in substantially identical functional form, in a corresponding encoder. For this reason, the disclosed subject matter focuses on decoder operation. The description of encoder technologies can be abbreviated as they are the inverse of the comprehensively described decoder technologies. Only in certain areas a more detail description is required and provided below.

During operation, in some examples, the source coder (**530**) may perform motion compensated predictive coding, which codes an input picture predictively with reference to one or more previously coded picture from the video sequence that were designated as "reference pictures." In this manner, the coding engine (**532**) codes differences between pixel blocks of an input picture and pixel blocks of reference picture(s) that may be selected as prediction reference(s) to the input picture.

The local video decoder (**533**) may decode coded video data of pictures that may be designated as reference pictures, based on symbols created by the source coder (**530**). Operations of the coding engine (**532**) may advantageously be lossy processes. When the coded video data may be decoded at a video decoder (not shown in FIG. **5**), the reconstructed video sequence typically may be a replica of the source video sequence with some errors. The local video decoder (**533**) replicates decoding processes that may be performed by the video decoder on reference pictures and may cause reconstructed reference pictures to be stored in the reference picture cache (**534**). In this manner, the video encoder (**503**) may store copies of reconstructed reference pictures locally that have common content as the reconstructed reference pictures that will be obtained by a far-end video decoder (absent transmission errors).

The predictor (**535**) may perform prediction searches for the coding engine (**532**). That is, for a new picture to be coded, the predictor (**535**) may search the reference picture memory (**534**) for sample data (as candidate reference pixel blocks) or certain metadata such as reference picture motion vectors, block shapes, and so on, that may serve as an appropriate prediction reference for the new pictures. The predictor (**535**) may operate on a sample block-by-pixel block basis to find appropriate prediction references. In some cases, as determined by search results obtained by the predictor (**535**), an input picture may have prediction references drawn from multiple reference pictures stored in the reference picture memory (**534**).

The controller (**550**) may manage coding operations of the source coder (**530**), including, for example, setting of parameters and subgroup parameters used for encoding the video data.

Output of all aforementioned functional units may be subjected to entropy coding in the entropy coder (**545**). The entropy coder (**545**) translates the symbols as generated by the various functional units into a coded video sequence, by lossless compressing the symbols according to technologies such as Huffman coding, variable length coding, arithmetic coding, and so forth.

The transmitter (**540**) may buffer the coded video sequence(s) as created by the entropy coder (**545**) to prepare for transmission via a communication channel (**560**), which may be a hardware/software link to a storage device which would store the encoded video data. The transmitter (**540**) may merge coded video data from the video coder (**503**) with other data to be transmitted, for example, coded audio data and/or ancillary data streams (sources not shown).

The controller (**550**) may manage operation of the video encoder (**503**). During coding, the controller (**550**) may assign to each coded picture a certain coded picture type, which may affect the coding techniques that may be applied to the respective picture. For example, pictures often may be assigned as one of the following picture types:

An Intra Picture (I picture) may be one that may be coded and decoded without using any other picture in the sequence as a source of prediction. Some video codecs allow for different types of intra pictures, including, for example Independent Decoder Refresh ("IDR") Pictures. A person skilled in the art is aware of those variants of I pictures and their respective applications and features.

A predictive picture (P picture) may be one that may be coded and decoded using intra prediction or inter prediction using at most one motion vector and reference index to predict the sample values of each block.

A bi-directionally predictive picture (B Picture) may be one that may be coded and decoded using intra prediction or inter prediction using at most two motion vectors and reference indices to predict the sample values of each block. Similarly, multiple-predictive pictures can use more than two reference pictures and associated metadata for the reconstruction of a single block.

Source pictures commonly may be subdivided spatially into a plurality of sample blocks (for example, blocks of 4×4, 8×8, 4×8, or 16×16 samples each) and coded on a block-by-block basis. Blocks may be coded predictively with reference to other (already coded) blocks as determined by the coding assignment applied to the blocks' respective pictures. For example, blocks of I pictures may be coded non-predictively or they may be coded predictively with reference to already coded blocks of the same picture (spatial prediction or intra prediction). Pixel blocks of P pictures may be coded predictively, via spatial prediction or via temporal prediction with reference to one previously coded reference picture. Blocks of B pictures may be coded predictively, via spatial prediction or via temporal prediction with reference to one or two previously coded reference pictures.

The video encoder (**503**) may perform coding operations according to a predetermined video coding technology or standard, such as ITU-T Rec. H.265. In its operation, the video encoder (**503**) may perform various compression operations, including predictive coding operations that exploit temporal and spatial redundancies in the input video sequence. The coded video data, therefore, may conform to a syntax specified by the video coding technology or standard being used.

In an embodiment, the transmitter (**540**) may transmit additional data with the encoded video. The source coder (**530**) may include such data as part of the coded video sequence. Additional data may comprise temporal/spatial/SNR enhancement layers, other forms of redundant data such as redundant pictures and slices, SEI messages, VUI parameter set fragments, and so on.

A video may be captured as a plurality of source pictures (video pictures) in a temporal sequence. Intra-picture prediction (often abbreviated to intra prediction) makes use of spatial correlation in a given picture, and inter-picture prediction makes uses of the (temporal or other) correlation between the pictures. In an example, a specific picture under encoding/decoding, which is referred to as a current picture, is partitioned into blocks. When a block in the current picture is similar to a reference block in a previously coded and still buffered reference picture in the video, the block in the current picture can be coded by a vector that is referred to as a motion vector. The motion vector points to the reference block in the reference picture, and can have a third dimension identifying the reference picture, in case multiple reference pictures are in use.

In some embodiments, a bi-prediction technique can be used in the inter-picture prediction. According to the bi-prediction technique, two reference pictures, such as a first reference picture and a second reference picture that are both prior in decoding order to the current picture in the video (but may be in the past and future, respectively, in display order) are used. A block in the current picture can be coded by a first motion vector that points to a first reference block in the first reference picture, and a second motion vector that points to a second reference block in the second reference picture. The block can be predicted by a combination of the first reference block and the second reference block.

Further, a merge mode technique can be used in the inter-picture prediction to improve coding efficiency.

According to some embodiments of the disclosure, predictions, such as inter-picture predictions and intra-picture predictions are performed in the unit of blocks. For example, according to the HEVC standard, a picture in a sequence of video pictures is partitioned into coding tree units (CTU) for compression, the CTUs in a picture have the same size, such as 64×64 pixels, 32×32 pixels, or 16×16 pixels. In general, a CTU includes three coding tree blocks (CTBs), which are one luma CTB and two chroma CTBs. Each CTU can be recursively quad-tree split into one or multiple CUs. For example, a CTU of 64×64 pixels can be split into one CU of 64×64 pixels, or 4 CUs of 32×32 pixels, or 16 CUs of 16×16 pixels. In an example, each CU is analyzed to determine a prediction type for the CU, such as an inter prediction type or an intra prediction type. The CU is split into one or more prediction units (PUs) depending on the temporal and/or spatial predictability. Generally, each PU includes a luma prediction block (PB), and two chroma PBs. In an embodiment, a prediction operation in coding (encoding/decoding) is performed in the unit of a prediction block. Using a luma prediction block as an example of a prediction block, the prediction block includes a matrix of values (e.g., luma values) for pixels, such as 8×8 pixels, 16×16 pixels, 8×16 pixels, 16×8 pixels, and the like.

FIG. **6** shows a diagram of a video encoder (**603**) according to another embodiment of the disclosure. The video encoder (**603**) is configured to receive a processing block (e.g., a prediction block) of sample values within a current video picture in a sequence of video pictures, and encode the processing block into a coded picture that is part of a coded video sequence. In an example, the video encoder (**603**) is used in the place of the video encoder (**303**) in the FIG. **3** example.

In an HEVC example, the video encoder (**603**) receives a matrix of sample values for a processing block, such as a prediction block of 8×8 samples, and the like. The video encoder (**603**) determines whether the processing block is best coded using intra mode, inter mode, or bi-prediction mode using, for example, rate-distortion optimization. When the processing block is to be coded in intra mode, the video encoder (**603**) may use an intra prediction technique to

encode the processing block into the coded picture; and when the processing block is to be coded in inter mode or bi-prediction mode, the video encoder (**603**) may use an inter prediction or bi-prediction technique, respectively, to encode the processing block into the coded picture. In certain video coding technologies, merge mode can be an inter picture prediction submode where the motion vector is derived from one or more motion vector predictors without the benefit of a coded motion vector component outside the predictors. In certain other video coding technologies, a motion vector component applicable to the subject block may be present. In an example, the video encoder (**603**) includes other components, such as a mode decision module (not shown) to determine the mode of the processing blocks.

In the FIG. **6** example, the video encoder (**603**) includes the inter encoder (**630**), an intra encoder (**622**), a residue calculator (**623**), a switch (**626**), a residue encoder (**624**), a general controller (**621**), and an entropy encoder (**625**) coupled together as shown in FIG. **6**.

The inter encoder (**630**) is configured to receive the samples of the current block (e.g., a processing block), compare the block to one or more reference blocks in reference pictures (e.g., blocks in previous pictures and later pictures), generate inter prediction information (e.g., description of redundant information according to inter encoding technique, motion vectors, merge mode information), and calculate inter prediction results (e.g., predicted block) based on the inter prediction information using any suitable technique. In some examples, the reference pictures are decoded reference pictures that are decoded based on the encoded video information.

The intra encoder (**622**) is configured to receive the samples of the current block (e.g., a processing block), in some cases compare the block to blocks already coded in the same picture, generate quantized coefficients after transform, and in some cases also intra prediction information (e.g., an intra prediction direction information according to one or more intra encoding techniques). In an example, the intra encoder (**622**) also calculates intra prediction results (e.g., predicted block) based on the intra prediction information and reference blocks in the same picture.

The general controller (**621**) is configured to determine general control data and control other components of the video encoder (**603**) based on the general control data. In an example, the general controller (**621**) determines the mode of the block, and provides a control signal to the switch (**626**) based on the mode. For example, when the mode is the intra mode, the general controller (**621**) controls the switch (**626**) to select the intra mode result for use by the residue calculator (**623**), and controls the entropy encoder (**625**) to select the intra prediction information and include the intra prediction information in the bitstream; and when the mode is the inter mode, the general controller (**621**) controls the switch (**626**) to select the inter prediction result for use by the residue calculator (**623**), and controls the entropy encoder (**625**) to select the inter prediction information and include the inter prediction information in the bitstream.

The residue calculator (**623**) is configured to calculate a difference (residue data) between the received block and prediction results selected from the intra encoder (**622**) or the inter encoder (**630**). The residue encoder (**624**) is configured to operate based on the residue data to encode the residue data to generate the transform coefficients. In an example, the residue encoder (**624**) is configured to convert the residue data from a spatial domain to a frequency domain, and generate the transform coefficients. The transform coefficients are then subject to quantization processing to obtain quantized transform coefficients. In various embodiments, the video encoder (**603**) also includes a residue decoder (**628**). The residue decoder (**628**) is configured to perform inverse-transform, and generate the decoded residue data. The decoded residue data can be suitably used by the intra encoder (**622**) and the inter encoder (**630**). For example, the inter encoder (**630**) can generate decoded blocks based on the decoded residue data and inter prediction information, and the intra encoder (**622**) can generate decoded blocks based on the decoded residue data and the intra prediction information. The decoded blocks are suitably processed to generate decoded pictures and the decoded pictures can be buffered in a memory circuit (not shown) and used as reference pictures in some examples.

The entropy encoder (**625**) is configured to format the bitstream to include the encoded block. The entropy encoder (**625**) is configured to include various information according to a suitable standard, such as the HEVC standard. In an example, the entropy encoder (**625**) is configured to include the general control data, the selected prediction information (e.g., intra prediction information or inter prediction information), the residue information, and other suitable information in the bitstream. Note that, according to the disclosed subject matter, when coding a block in the merge submode of either inter mode or bi-prediction mode, there is no residue information.

FIG. **7** shows a diagram of a video decoder (**710**) according to another embodiment of the disclosure. The video decoder (**710**) is configured to receive coded pictures that are part of a coded video sequence, and decode the coded pictures to generate reconstructed pictures. In an example, the video decoder (**710**) is used in the place of the video decoder (**310**) in the FIG. **3** example.

In the FIG. **7** example, the video decoder (**710**) includes an entropy decoder (**771**), an inter decoder (**780**), a residue decoder (**773**), a reconstruction module (**774**), and an intra decoder (**772**) coupled together as shown in FIG. **7**.

The entropy decoder (**771**) can be configured to reconstruct, from the coded picture, certain symbols that represent the syntax elements of which the coded picture is made up. Such symbols can include, for example, the mode in which a block is coded (such as, for example, intra mode, inter mode, bi-predicted mode, the latter two in merge submode or another submode), prediction information (such as, for example, intra prediction information or inter prediction information) that can identify certain sample or metadata that is used for prediction by the intra decoder (**772**) or the inter decoder (**780**), respectively, residual information in the form of, for example, quantized transform coefficients, and the like. In an example, when the prediction mode is inter or bi-predicted mode, the inter prediction information is provided to the inter decoder (**780**); and when the prediction type is the intra prediction type, the intra prediction information is provided to the intra decoder (**772**). The residual information can be subject to inverse quantization and is provided to the residue decoder (**773**).

The inter decoder (**780**) is configured to receive the inter prediction information, and generate inter prediction results based on the inter prediction information.

The intra decoder (**772**) is configured to receive the intra prediction information, and generate prediction results based on the intra prediction information.

The residue decoder (**773**) is configured to perform inverse quantization to extract de-quantized transform coefficients, and process the de-quantized transform coefficients to convert the residual from the frequency domain to the spatial domain. The residue decoder (**773**) may also require

certain control information (to include the Quantizer Parameter (QP)), and that information may be provided by the entropy decoder (**771**) (data path not depicted as this may be low volume control information only).

The reconstruction module (**774**) is configured to combine, in the spatial domain, the residual as output by the residue decoder (**773**) and the prediction results (as output by the inter or intra prediction modules as the case may be) to form a reconstructed block, that may be part of the reconstructed picture, which in turn may be part of the reconstructed video. It is noted that other suitable operations, such as a deblocking operation and the like, can be performed to improve the visual quality.

It is noted that the video encoders (**303**), (**503**), and (**603**), and the video decoders (**310**), (**410**), and (**710**) can be implemented using any suitable technique. In an embodiment, the video encoders (**303**), (**503**), and (**603**), and the video decoders (**310**), (**410**), and (**710**) can be implemented using one or more integrated circuits. In another embodiment, the video encoders (**303**), (**503**), and (**603**), and the video decoders (**310**), (**410**), and (**710**) can be implemented using one or more processors that execute software instructions.

II. HEVC Block Partition Structure

A CTU, such as in HEVC, can be split into CUs by using a quad-tree structure denoted as a coding tree to adapt to various local characteristics. The decision on whether to code a picture area using inter-picture (temporal) or intra-picture (spatial) prediction can be made at the CU level. Each CU can be further split into one, two, or four PUs according to a PU splitting type. Inside one PU, the same prediction process is applied and the relevant information is transmitted to a decoder on a PU basis. After obtaining a residual block by applying the prediction process based on the PU splitting type, a CU can be partitioned into TUs according to another QT structure like the coding tree for the CU.

One feature of the HEVC structure is that it has multiple partition conceptions including CU, PU, and TU. In HEVC, a CU or a TU can be limited to a square shape, while a PU may be a square or rectangular shape for an inter predicted block. Rectangular shape PUs for intra prediction and transform that were proposed can be extended to be used in joint exploration model (JEM).

III. Block Partitioning Structure Using QTBT

A quad-tree (QT) plus binary-tree (BT) plus ternary tree (TT) partitioning structure can be applied, such as in VVC test model (VTM). The quad-tree plus binary tree (QTBT) structure removes the concepts of multiple partition types (i.e., it removes the separation of the CU, PU, and TU concepts), and supports more flexibility for CU partition shapes.

In the QTBT structure, a CU can have either a square or rectangular shape. As shown in FIGS. **8**A and **8**B, a CTU is first partitioned by a QT structure. The QT leaf nodes can be further partitioned by a BT structure. There are two splitting types, symmetric horizontal splitting and symmetric vertical splitting, in the BT splitting. The BT leaf nodes are CUs, and segmentation into two CUs is used for prediction and transform processing without any further partitioning. Accordingly, a CU, PU, and TU can have the same block size in the QTBT structure.

A CU sometimes can include CBs of different color components, such as in JEM. For example, one CU can contain one luma CB and two chroma CBs in the case of P and B slices with the 4:2:0 chroma format. In other examples, a CU can include CBs of a single component, e.g., one CU can contain only one luma CB or just two chroma CBs in the case of I slices.

The following parameters are defined for the QTBT partitioning scheme:

CTU size: the root node size of a QT, for example as in HEVC

MinQTSize: the minimum allowed QT leaf node size

MaxBTSize: the maximum allowed BT root node size

MaxBTDepth: the maximum allowed BT depth

MinBTSize: the minimum allowed BT leaf node size

In one example of the QTBT partitioning structure, the CTU size is set as 128×128 luma samples with two corresponding 64×64 blocks of chroma samples, the MinQTSize is set as 16×16, the MaxBTSize is set as 64×64, the MinBTSize (for both width and height) is set as 4×4, and the MaxBTDepth is set as 4. The QT partitioning is applied to the CTU first to generate QT leaf nodes. The QT leaf nodes may have a size from 16×16 (i.e., the MinQTSize) to 128×128 (i.e., the CTU size). If the leaf QT node is 128×128, it will not be further split by the BT since the size exceeds the MaxBTSize (i.e., 64×64). Otherwise, the leaf QT node could be further partitioned by the BT tree. Therefore, the QT leaf node is also the root node for the BT and it has a BT depth of 0. When the BT depth reaches MaxBTDepth (i.e., 4), no further splitting is considered. When the BT node has a width equal to MinBTSize (i.e., 4), no further horizontal splitting is considered. Similarly, when the BT node has a height equal to MinBTSize, no further vertical splitting is considered. The leaf nodes of the BT are further processed by prediction and transform processing without any further partitioning. For example, the maximum CTU size is 256×256 luma samples such as in JEM.

FIG. **8**A illustrates an example of a block partition (**8100**) by using a QTBT partitioning structure (**8200**) and FIG. **8**B illustrates the corresponding QTBT structure (**8200**). The solid lines indicate QT splits and the dotted lines indicate binary tree BT splits. In each non-leaf BT split node, a flag is signaled to indicate a splitting type (i.e., a symmetric horizontal split or a symmetric vertical split). For example, in the FIG. **8**B example, "0" indicates a symmetric horizontal split and "1" indicates a symmetric vertical split. For a QT split, however, a split type flag is not indicated or signaled because the QT split splits a non-leaf node both horizontally and vertically to produce four smaller blocks with an equal size.

Referring to FIG. **8**B, in the QTBT structure (**8200**), a root node (**8201**) is first partitioned by a QT structure into QT nodes (**8211**)-(**8214**). Accordingly, as shown in FIG. **8**A, a coding tree block (**8101**) is partitioned into four equal size blocks (**8111**)-(**8114**) by the solid lines.

Referring back to FIG. **8**B, the QT nodes (**8211**) and (**8212**) are further split by two BT splits, respectively. As mentioned above, a BT split includes two splitting types (i.e., a symmetric horizontal split and a symmetric vertical split). The BT split for the non-leaf QT node (**8211**) is indicated as "1" and thus the non-leaf QT node (**8211**) can be split into two nodes (**8221**) and (**8222**) by using a symmetric vertical split. The BT split for the non-leaf QT node (**8212**) is indicated as "0" and thus the non-leaf QT node (**8212**) can be split into two nodes (**8223**) and (**8224**) by using a symmetric horizontal split. The non-leaf QT node (**8213**) is further split into four nodes (**8225**)-(**8228**) by another QTBT structure. The node (**8214**) is not further split and thus a leaf node. Accordingly, as shown in FIG. **8**A, the block (**8111**) is vertically partitioned into two equal size blocks (**8121**) and (**8122**), the block (**8112**) is horizontally

partitioned into two equal size blocks, the block (**8113**) is partitioned into four equal size blocks, and the block (**8114**) is not further partitioned.

Referring back to FIG. **8**B, in deeper levels, some of the nodes, for example, the nodes (**8221**)-(**8228**) are further split while others are not. For example, the non-leaf BT node (**8221**) is further split into two leaf nodes (**8231**) and (**8232**) by a symmetric vertical split while the leaf node (**8222**) is not further split. Accordingly, as shown in FIG. **8**A, the block (**8121**) is partitioned into two equal size blocks (**8131**) and (**8132**) while the block (**8122**) is not further partitioned.

After splitting of the QTBT structure (**8200**) is completed, leaf nodes that are not further split are CUs that are used for prediction and transform processing. Thus, a CU, a PU that is associated with the CU, and a TU that is associated with the CU can have a same block size in the QTBT structure. In addition, in the QTBT structure, a CU can include CBs in different color components. For example, in a 4:2:0 format, one CU can include one luma CB and two chroma CBs in a P or B slice. However, in some other embodiments, a CU may include CBs in a single component. For example, in an I slice, one CU may include one luma CB or two chroma CBs. That is to say, the QTBT structure supports an ability for luma and chroma to have different partitioning structures.

Currently, for P and B slices, the luma and chroma CTBs in one CTU share the same QTBT structure. However, for I slices, the luma CTB is partitioned into CUs by a QTBT structure and the chroma CTBs are partitioned into chroma CUs by another QTBT structure. Accordingly, a CU in an I slice includes a coding block of the luma component or coding blocks of two chroma components, and a CU in a P or B slice includes coding blocks of all three color components.

Inter prediction for small blocks is restricted to reduce the memory access of motion compensation, for example in HEVC, such that bi-prediction is not supported for 4×8 and 8×4 blocks, and inter prediction is not supported for 4×4 blocks. In the QTBT as implemented in the JEM-7.0, these restrictions are removed.

For example, in VTM6, the coding tree scheme supports the ability for the luma and chroma to have separate block tree structures. In an example, for P and B slices, the luma and chroma CTBs in one CTU have to share the same coding tree structure. However, for I slices, the luma and chroma can have separate block tree structures. When separate block tree mode is applied, luma CTB is partitioned into CUs by one coding tree structure, and the chroma CTBs are partitioned into chroma CUs by another coding tree structure. This means that a CU in an I slice may include a coding block of the luma component or coding blocks of two chroma components, and a CU in a P or B slice always include coding blocks of all three color components unless the video is monochrome.

IV. Block Partitioning Structure Using Triple-Trees (TT)

In addition to the QTBT structure described above, another splitting structure called multi-type-tree (MTT) structure can be more flexible than the QTBT structure. In MTT, other than QT and BT, horizontal and vertical center-side TTs are introduced, as shown in FIG. **9**A and FIG. **9**B.

FIG. **9**A shows an example of vertical center-side TT partitioning. For example, a block (**910**) is vertically split into three sub-blocks (**911**)-(**913**) where the sub-block (**912**) is located in the middle of the block (**910**).

FIG. **9**B shows an example of horizontal center-side TT partitioning. For example, a block (**920**) is horizontally split into three sub-blocks (**921**)-(**923**) where the sub-block (**922**) is located in the middle of the block (**920**).

Similar to a BT split, in a TT split, a flag is signaled to indicate a splitting type (i.e., a symmetric horizontal split or a symmetric vertical split). In an example, "0" indicates a symmetric horizontal split and "1" indicates a symmetric vertical split.

One benefit of the TT partitioning is that the TT partitioning can be a complement to the QT partitioning and the BT partitioning. For example, the TT partitioning is able to capture objects which are located in a block center while the QT partitioning and the BT partitioning are always splitting along the block center. Another benefit of the TT partitioning is that the width and height of the partitions through the TT partitioning are always a power of 2, so that no additional transforms are needed.

A MTT structure, including quad-tree, binary-tree, and ternary-tree splitting types, can be referred to as a QTBTTT structure. Similar to a QTBT structure, a QTBTTT structure also supports luma and chroma having different structures. For example, in an I slice, a QTBTTT structure used to partition a luma CTB can be different from a QTBTTT structure used to partition a chroma CTB. This means that when a separated tree structure is enabled, a CU includes one luma CB or two chroma CBs. However, in a P or B slice, a luma CTB can share the same QTBTTT structure with a chroma CTB in one CTU. This means that when the separated tree structure is disabled, a CU includes all three CBs (i.e., one luma CB and two chroma CBs).

Dual-tree or separate-tree can be used for I-slice, such as in VVC. That is, one tree is used for a luma component and the other tree is used for a chroma component. For a B-slice and a P-slice, one single-tree can be shared by both luma and chroma components.

The design of a two-level tree is can be motivated by complexity reduction. In an example, the complexity of traversing a tree is T^D, where T denotes the number of split types, and D is the depth of tree.

V. Sub-Block Transform (SBT)

Spatially varying transform (SVT) can also be referred to as a sub-block transform (SBT). The SBT can be applied to inter prediction residuals. For example, a coding block can be partitioned into sub-blocks, and only part of the sub-blocks is treated as a residual block. Zero residual is assumed for the remaining part of the sub-blocks. Therefore, the residual block is smaller than the coding block, and a transform size in SBT is smaller than the coding block size. For the region which is not covered by the residual block, no transform processing is performed.

FIGS. **10**A-**10**D show exemplary SBT modes according to some embodiments of the disclosure. The SBT modes support different SBT types such as SVT-H and SVT-V (e.g., vertically or horizontally partitions), sizes, and positions (e.g., left half, left quarter, right half, right quarter, top half, top quarter, bottom half, bottom quarter). The shaded regions labeled by letter "A" correspond to residual blocks with transforms, and the other regions can be assumed to be zero residual without transform.

FIGS. **11**-**15** show exemplary syntax related to SBT methods according to some embodiments of the disclosure. It can be seen that the SBT methods require one or more overhead bits (e.g., cu_sbt_flag, cu_sbt_quad_flag, cu_sbt_horizontal_flag, cu_sbt_pos_flag, etc.) to be signaled to indicate the SBT type or direction (e.g., horizontal or vertical), size (e.g., half or quarter), and position (e.g., left or right, top or bottom).

Specifically, in FIG. **11**, when sps_sbt_enable_flag is equal to 0, it specifies that SBT for the inter-predicted CU is disabled. When sps_sbt_enable_flag is equal to 1, it specifies that SBT for the inter-predicted CU is enabled.

In FIG. **12**, when slice_max_sbt_size_64_flag is equal to 0, it specifies that the maximum CU width and height for allowing SBT is 32. When slice_max_sbt_size_64_flag is equal to 1, it specifies that the maximum CU width and height for allowing SBT is 64. That is, maxSbtSize=slice_max_sbt_size_64_flag? 64:32.

In FIG. **13**, when cu_sbt_flag[x0][y0] is equal to 1, it specifies that SBT is used for the current CU. When cu_sbt_flag[x0][y0] is equal to 0, it specifies that SBT is not used for the current CU.

When cu_sbt_flag[x0][y0] is not present, its value is inferred to be equal to 0.

It is noted that when SBT is used, a CU is tiled into two TUs, one TU has residuals, and the other TU does not have residuals.

When cu_sbt_quad_flag[x0][y0] is equal to 1, it specifies that for the current CU, the SBT includes a TU of ¼ size of the current CU. When cu_sbt_quad_flag[x0][y0] is equal to 0, it specifies that for the current CU the SBT includes a TU of ½ size of the current CU.

When cu_sbt_quad_flag[x0][y0] is not present, its value is inferred to be equal to 0.

When cu_sbt_horizontal_flag[x0][y0] is equal to 1, it specifies that the current CU is tiled into 2 TUs by a horizontal split. When cu_sbt_horizontal_flag[x0][y0] is equal to 0, it specifies that the current CU is tiled into 2 TUs by a vertical split.

When cu_sbt_horizontal_flag[x0][y0] is not present, its value can be derived as follows: if cu_sbt_quad_flag[x0][y0] is equal to 1, cu_sbt_horizontal_flag[x0][y0] is set to be equal to allowSbtHoriQuad; otherwise (cu_sbt_quad_flag [x0][y0] is equal to 0), cu_sbt_horizontal_flag[x0][y0] is set to be equal to allowSbtHoriHalf.

When cu_sbt_pos_flag[x0][y0] is equal to 1, it specifies that the tu_cbf_luma, tu_cbf_cb, and tu_cbf_cr of the first TU in the current CU are not present in the bitstream. When cu_sbt_pos_flag[x0][y0] is equal to 0, it specifies that the tu_cbf_luma, tu_cbf_cb, and tu_cbf_cr of the second transform unit in the current CU are not present in the bitstream.

VI. Interaction Between CU Partitions and SBT

The present disclosure includes embodiments for improving performance of CU partitions and SBT.

It can be seen in the above discussion that a partition associated with an SBT mode can overlap or correspond to a partition associated with a CU partitioning structure. For example, an SBT mode can be performed on a CU with a size of 16×16 and the left 16×8 sub-CU may be transformed. This process can be mimicked or achieved by a further CU split, after which the residual coding can be perfumed on the left 16×8 sub-CU while the right 16×8 sub-CU has no residual.

This disclosure includes methods for disallowing an SBT mode for a current block if a partition associated with the SBT mode can be achieved by a partition associated with a CU partitioning structure. In some embodiments, when one or more SBT modes (e.g., a vertical SBT mode, a horizontal SBT mode, a half SBT mode, and/or a quarter SBT mode) are not allowed, the related syntax may be inferred without signaling.

The following embodiments are described using a block width of the current block and a vertical SBT mode as an example. The embodiments can be extended to the case of a block height and a horizontal SBT mode. In the following embodiments, a half SBT mode refers to an SBT mode that partitions one CU into two equal-size sub-TUs, and a quarter SBT mode refers to an SBT mode that partitions one CU into a sub-TU with one-quarter size and a sub-TU with three-quarter size. The sub-TU with one-quarter size may have residuals and the sub-TU with three-quarter size may not have any residual. In addition, the term block may be interpreted as a PB, a coding block, or a CU.

In one embodiment, the vertical SBT mode is not allowed for a block having a width that is greater than a threshold when a partitioning depth associated with the block is less than a maximum allowed partitioning depth associated with the block. In this case, the partition associated with the vertical SBT mode can be mimicked or achieved by one vertical partition, such as vertical BT or TT splits. In one example, the threshold is 8 for the half SBT mode if BT is allowed for the block. In another example, the threshold is 16 for the quarter SBT mode if TT is allowed for the block.

In one embodiment, when SBT mode partitioning can lead to small chroma blocks, such as 4×2, 2×4, and 2×2 chroma blocks, the SBT mode is not allowed.

In one embodiment, if only one SBT direction is allowed, a flag indicating the other SBT direction is not signaled but can be inferred. For example, cu_sbt_horizontal_flag is not signaled but can be inferred.

In one embodiment, if only some SBT sizes are allowed, a flag indicating an SBT mode having a minimum allowed SBT size that is larger than that of another SBT mode is not signaled but can be inferred. For example, if both of the half and quarter SBT modes are not allowed, cu_sbt_quad_flag is not signaled but can be inferred.

In one embodiment, if no SBT mode is allowed, a flag (e.g., cu_sbt_flag) indicating whether an SBT mode is performed on the block is not signaled but can be inferred.

In one embodiment, when a CU partition associated with a block can align with a sub-TU partition of a parent CU of the block by using an SBT mode, a transform type for the block can be determined based on the SBT mode. For example, if the block is generated based on a vertical BT split that partitions the parent CU into a left sub-CU (i.e., the block) and a right sub-CU, and the parent CU can be also partitioned using a vertical half SBT mode into a left sub-TU and a right sub-TU, then the block can be considered to be aligned with the left sub-TU, and the transform type of the block can be determined based on a transform type for the left sub-TU. For example, the transform type of the block can be found in a look-up transform table used for SBT mode based on a size of the block. In addition, in an example, a flag or index can be signaled to indicate whether the transform type of the block is DCT-2 or the same as the transform type used for the sub-TU partition of the parent CU.

According to aspects of the disclosure, context models (e.g., two context models) can be used for entropy coding the SBT mode information when related SBT modes are allowed.

In one embodiment, the context model selection for the flag cu_sbt_flag may be based on whether the partitioning depth (e.g., current BT depth+1) associated with the block is less than the maximum allowed parititoning depth (e.g., maximum allowed BT/TT depth).

In one embodiment, the context model selection for the flag cu_sbt_quad_flag may be based on the flag cu_sbt_horizontal_flag and the block width/height. For example, two context models for the flag cu_sbt_quad_flag may be used based on whether the following condition is true: (cu_sbt_horizontal_flag && height>=16)||(!cu_sbt_horizontal_flag

&& width>=16). When the condition is true, one context model is selected. When the condition is false, the other context mode is selected.

According to aspects of the disclosure, SBT can be disallowed, for example according to one or more of the above-described conditions, for certain types of CUs. In one embodiment, diallowing SBT according to one or more of the above-described conditions can only be applied to a non-merge inter block.

VII. Flowchart

FIG. **16** shows a flow chart outlining an exemplary process (**1600**) according to an embodiment of the disclosure. In various embodiments, the process (**1600**) is executed by processing circuitry, such as the processing circuitry in the terminal devices (**210**), (**220**), (**230**) and (**240**), the processing circuitry that performs functions of the video encoder (**303**), the processing circuitry that performs functions of the video decoder (**310**), the processing circuitry that performs functions of the video decoder (**410**), the processing circuitry that performs functions of the intra prediction module (**452**), the processing circuitry that performs functions of the video encoder (**503**), the processing circuitry that performs functions of the predictor (**535**), the processing circuitry that performs functions of the intra encoder (**622**), the processing circuitry that performs functions of the intra decoder (**772**), and the like. In some embodiments, the process (**1600**) is implemented in software instructions, thus when the processing circuitry executes the software instructions, the processing circuitry performs the process (**1600**).

The process (**1600**) may generally start at step (S**1610**), where the process (**1600**) decodes prediction information of a current block of a coding unit tree in a coded bit stream. The prediction information indicates at least one allowed block partitioning structure for the current block. Then, the process (**1600**) proceeds to step (S**1620**).

At step (S**1620**), the process (**1600**) determines a sub-block transform (SBT) mode for the current block based on the prediction information indicating that SBT is used for the current block. A partition of the current block based on the SBT mode is different from a partition of the current block based on the at least one allowed block partitioning structure. Then, the process (**1600**) proceeds to step (S**1630**).

At step (S**1630**), the process (**1600**) reconstructs the current block based on the SBT mode. Then, the process (**1600**) terminates.

In an embodiment, the process (**1600**) determines the SBT mode for the current block when (i) one of a width and a height of the current block is greater than a first threshold and (ii) a partitioning depth associated with the current block is less than a maximum allowed partitioning depth of the coding unit tree. In an example, the SBT mode is determined not to be a vertical SBT mode when the width of the current block is greater than the first threshold. In an example, the SBT mode is determined not to be a horizontal SBT mode when the height of the current block is greater than the first threshold. In an example, the SBT mode is determined not to be a half SBT mode when (i) the first threshold is 8 and (ii) the at least one allowed partitioning structure includes a binary tree partitioning structure. In an example, the SBT mode is determined not to be a quarter SBT mode when (i) the first threshold is 16 and (ii) the at least one allowed partitioning structure includes a triple tree partitioning structure.

In an embodiment, the process (**1600**) determines the SBT mode for the current block when (i) the current block is a chroma block and (ii) sizes of a plurality of sub-blocks of the current block are greater than a second threshold. The current block can be partitioned into the plurality of sub-blocks based on the SBT mode.

In an embodiment, a flag indicating a direction of the SBT mode is not included in the prediction information.

In an embodiment, a flag indicating a size of the SBT mode is not included in the prediction information.

In an embodiment, a flag indicating whether SBT is used for the current block is not included in the prediction information.

In an embodiment, the process (**1600**) determines a context model for a flag based on a partitioning depth associated with the current block. The flag indicates whether SBT is used for the current block.

In an embodiment, the process (**1600**) determines a context model for a flag based on at least one of a direction of the SBT mode, the width of the current block, and the height of the current block. The flag indicates a size of the SBT mode.

In an embodiment, the prediction information indicates that the current block is a non-merge inter block.

VIII. Computer System

The presented methods may be used separately or combined in any order. Further, each of the embodiments, encoder, and decoder may be implemented by processing circuitry (e.g., one or more processors or one or more integrated circuits). In one example, the one or more processors execute a program that is stored in a non-transitory computer-readable medium.

The techniques described above, can be implemented as computer software using computer-readable instructions and physically stored in one or more computer-readable media. For example, FIG. **17** shows a computer system (**1700**) suitable for implementing certain embodiments of the disclosed subject matter.

The computer software can be coded using any suitable machine code or computer language, that may be subject to assembly, compilation, linking, or like mechanisms to create code comprising instructions that can be executed directly, or through interpretation, micro-code execution, and the like, by one or more computer central processing units (CPUs), Graphics Processing Units (GPUs), and the like.

The instructions can be executed on various types of computers or components thereof, including, for example, personal computers, tablet computers, servers, smartphones, gaming devices, internet of things devices, and the like.

The components shown in FIG. **17** for computer system (**1700**) are exemplary in nature and are not intended to suggest any limitation as to the scope of use or functionality of the computer software implementing embodiments of the present disclosure. Neither should the configuration of components be interpreted as having any dependency or requirement relating to any one or combination of components illustrated in the exemplary embodiment of a computer system (**1700**).

Computer system (**1700**) may include certain human interface input devices. Such a human interface input device may be responsive to input by one or more human users through, for example, tactile input (such as: keystrokes, swipes, data glove movements), audio input (such as: voice, clapping), visual input (such as: gestures), olfactory input (not depicted). The human interface devices can also be used to capture certain media not necessarily directly related to conscious input by a human, such as audio (such as: speech, music, ambient sound), images (such as: scanned images, photographic images obtain from a still image camera),

video (such as two-dimensional video, three-dimensional video including stereoscopic video).

Input human interface devices may include one or more of (only one of each depicted): keyboard (**1701**), mouse (**1702**), trackpad (**1703**), touch screen (**1710**), data-glove (not shown), joystick (**1705**), microphone (**1706**), scanner (**1707**), camera (**1708**).

Computer system (**1700**) may also include certain human interface output devices. Such human interface output devices may be stimulating the senses of one or more human users through, for example, tactile output, sound, light, and smell/taste. Such human interface output devices may include tactile output devices (for example tactile feedback by the touch-screen (**1710**), data-glove (not shown), or joystick (**1705**), but there can also be tactile feedback devices that do not serve as input devices), audio output devices (such as: speakers (**1709**), headphones (not depicted)), visual output devices (such as screens (**1710**) to include CRT screens, LCD screens, plasma screens, OLED screens, each with or without touch-screen input capability, each with or without tactile feedback capability—some of which may be capable to output two dimensional visual output or more than three dimensional output through means such as stereographic output; virtual-reality glasses (not depicted), holographic displays and smoke tanks (not depicted)), and printers (not depicted). These visual output devices (such as screens (**1710**)) can be connected to a system bus (**1748**) through a graphics adapter (**1750**).

Computer system (**1700**) can also include human accessible storage devices and their associated media such as optical media including CD/DVD ROM/RW (**1720**) with CD/DVD or the like media (**1721**), thumb-drive (**1722**), removable hard drive or solid state drive (**1723**), legacy magnetic media such as tape and floppy disc (not depicted), specialized ROM/ASIC/PLD based devices such as security dongles (not depicted), and the like.

Those skilled in the art should also understand that term "computer readable media" as used in connection with the presently disclosed subject matter does not encompass transmission media, carrier waves, or other transitory signals.

Computer system (**1700**) can also include a network interface (**1754**) to one or more communication networks (**1755**). The one or more communication networks (**1755**) can for example be wireless, wireline, optical. The one or more communication networks (**1755**) can further be local, wide-area, metropolitan, vehicular and industrial, real-time, delay-tolerant, and so on. Examples of the one or more communication networks (**1755**) include local area networks such as Ethernet, wireless LANs, cellular networks to include GSM, 3G, 4G, 5G, LTE and the like, TV wireline or wireless wide area digital networks to include cable TV, satellite TV, and terrestrial broadcast TV, vehicular and industrial to include CANBus, and so forth. Certain networks commonly require external network interface adapters that attached to certain general purpose data ports or peripheral buses (**1749**) (such as, for example USB ports of the computer system (**1700**)); others are commonly integrated into the core of the computer system (**1700**) by attachment to a system bus as described below (for example Ethernet interface into a PC computer system or cellular network interface into a smartphone computer system). Using any of these networks, computer system (**1700**) can communicate with other entities. Such communication can be uni-directional, receive only (for example, broadcast TV), uni-directional send-only (for example CANbus to certain CANbus devices), or bi-directional, for example to other computer systems using local or wide area digital networks. Certain protocols and protocol stacks can be used on each of those networks and network interfaces as described above.

Aforementioned human interface devices, human-accessible storage devices, and network interfaces can be attached to a core (**1740**) of the computer system (**1700**).

The core (**1740**) can include one or more Central Processing Units (CPU) (**1741**), Graphics Processing Units (GPU) (**1742**), specialized programmable processing units in the form of Field Programmable Gate Areas (FPGA) (**1743**), hardware accelerators for certain tasks (**1744**), and so forth. These devices, along with Read-only memory (ROM) (**1745**), Random-access memory (**1746**), internal mass storage such as internal non-user accessible hard drives, SSDs, and the like (**1747**), may be connected through the system bus (**1748**). In some computer systems, the system bus (**1748**) can be accessible in the form of one or more physical plugs to enable extensions by additional CPUs, GPU, and the like. The peripheral devices can be attached either directly to the core's system bus (**1748**), or through a peripheral bus (**1749**). Architectures for a peripheral bus include PCI, USB, and the like.

CPUs (**1741**), GPUs (**1742**), FPGAs (**1743**), and accelerators (**1744**) can execute certain instructions that, in combination, can make up the aforementioned computer code. That computer code can be stored in ROM (**1745**) or RAM (**1746**). Transitional data can be also be stored in RAM (**1746**), whereas permanent data can be stored for example, in the internal mass storage (**1747**). Fast storage and retrieve to any of the memory devices can be enabled through the use of cache memory, that can be closely associated with one or more CPU (**1741**), GPU (**1742**), mass storage (**1747**), ROM (**1745**), RAM (**1746**), and the like.

The computer readable media can have computer code thereon for performing various computer-implemented operations. The media and computer code can be those specially designed and constructed for the purposes of the present disclosure, or they can be of the kind well known and available to those having skill in the computer software arts.

As an example and not by way of limitation, the computer system having architecture (**1700**), and specifically the core (**1740**) can provide functionality as a result of processor(s) (including CPUs, GPUs, FPGA, accelerators, and the like) executing software embodied in one or more tangible, computer-readable media. Such computer-readable media can be media associated with user-accessible mass storage as introduced above, as well as certain storage of the core (**1740**) that are of non-transitory nature, such as core-internal mass storage (**1747**) or ROM (**1745**). The software implementing various embodiments of the present disclosure can be stored in such devices and executed by core (**1740**). A computer-readable medium can include one or more memory devices or chips, according to particular needs. The software can cause the core (**1740**) and specifically the processors therein (including CPU, GPU, FPGA, and the like) to execute particular processes or particular parts of particular processes described herein, including defining data structures stored in RAM (**1746**) and modifying such data structures according to the processes defined by the software. In addition or as an alternative, the computer system can provide functionality as a result of logic hardwired or otherwise embodied in a circuit (for example: accelerator (**1744**)), which can operate in place of or together with software to execute particular processes or particular parts of particular processes described herein. Reference to software can encompass logic, and vice versa, where appropriate. Reference to a computer-readable media

can encompass a circuit (such as an integrated circuit (IC)) storing software for execution, a circuit embodying logic for execution, or both, where appropriate. The present disclosure encompasses any suitable combination of hardware and software.

While this disclosure has described several exemplary embodiments, there are alterations, permutations, and various substitute equivalents, which fall within the scope of the disclosure. It will thus be appreciated that those skilled in the art will be able to devise numerous systems and methods which, although not explicitly shown or described herein, embody the principles of the disclosure and are thus within the spirit and scope thereof.

APPENDIX A: ACRONYMS

AMT: Adaptive Multiple Transform
AMVP: Advanced Motion Vector Prediction
ASIC: Application-Specific Integrated Circuit
ATMVP: Alternative/Advanced Temporal Motion Vector Prediction
BDOF: Bi-directional Optical Flow
BDPCM (or RDPCM): Residual Difference Pulse Coded Modulation
BIO: Bi-directional Optical Flow
BMS: Benchmark Set
BT: Binary Tree
BV: Block Vector
CANBus: Controller Area Network Bus
CB: Coding Block
CBF: Coded Block Flag
CCLM: Cross-Component Linear Mode/Model
CD: Compact Disc
CPR: Current Picture Referencing
CPU: Central Processing Unit
CRT: Cathode Ray Tube
CTB: Coding Tree Block
CTU: Coding Tree Unit
CU: Coding Unit
DM: Derived Mode
DPB: Decoder Picture Buffer
DVD: Digital Video Disc
EMT: Enhanced Multiple Transform
FPGA: Field Programmable Gate Areas
GOP: Group of Picture
GPU: Graphics Processing Unit
GSM: Global System for Mobile communications
HDR: High Dynamic Range
HEVC: High Efficiency Video Coding
HRD: Hypothetical Reference Decoder
IBC: Intra Block Copy
IC: Integrated Circuit
IDT: Identify Transform
ISP: Intra Sub-Partitions
JEM: Joint Exploration Model
JVET: Joint Video Exploration Team
LAN: Local Area Network
LCD: Liquid-Crystal Display
LFNST: Low Frequency Non-Separable Transform, or Low Frequency Non-Separable Secondary Transform
LTE: Long-Term Evolution
L_CCLM: Left-Cross-Component Linear Mode/Model
LT_CCLM: Left and Top Cross-Component Linear Mode/Model
MIP: Matrix based Intra Prediction
MPM: Most Probable Mode
MRLP (or MRL): Multiple Reference Line Prediction
MTS: Multiple Transform Selection
MV: Motion Vector
NSST: Non-Separable Secondary Transform
OLED: Organic Light-Emitting Diode
PBs: Prediction Blocks
PCI: Peripheral Component Interconnect
PDPC: Position Dependent Prediction Combination
PLD: Programmable Logic Device
PPR: Parallel-Processable Region
PPS: Picture Parameter Set
PU: Prediction Unit
QT: Quad-Tree
RAM: Random Access Memory
ROM: Read-Only Memory
RST: Reduced-Size Transform
SBT: Sub-block Transform
SCC: Screen Content Coding
SCIPU: Small Chroma Intra Prediction Unit
SDR: Standard Dynamic Range
SEI: Supplementary Enhancement Information
SNR: Signal Noise Ratio
SPS: Sequence Parameter Set
SSD: Solid-state Drive
SVT: Spatially Varying Transform
TSM: Transform Skip Mode
TT: Ternary Tree
TU: Transform Unit
T_CCLM: Top Cross-Component Linear Mode/Model
USB: Universal Serial Bus
VPDU: Visual Process Data Unit
VPS: Video Parameter Set
VUI: Video Usability Information
VVC: Versatile Video Coding
WAIP: Wide-Angle Intra Prediction

What is claimed is:

1. A method for video encoding, comprising:

generating, by processing circuitry of a video encoder, prediction information of a current block of a coding unit tree, the prediction information indicating that at least one block partitioning structure is allowed for the current block and that a sub-block transform (SBT) mode is used for the current block;

determining, by the processing circuitry, a partition direction and a partition size of the SBT mode for the current block based on the at least one allowed block partitioning structure indicated in the prediction information and based on a comparison between at least one dimension of the current block and a first threshold, a partition of the current block based on the partition direction and the partition size of the SBT mode being different from at least one partition of the current block based on the at least one allowed block partitioning structure indicated in the prediction information; and

encoding, by the processing circuitry, the current block based on the determined partition direction and partition size of the SBT mode.

2. The method of claim **1**, wherein the determining includes:

determining the partition direction of the SBT mode for the current block based on (i) one of a width and a height of the current block being greater than the first threshold and (ii) a partitioning depth associated with the current block being less than a maximum allowed partitioning depth of the coding unit tree.

3. The method of claim **2**, wherein the determining includes at least one of:

determining that the partition direction of the SBT mode is not a vertical partition direction based on the width of the current block being greater than the first threshold; and

determining that the partition direction of the SBT mode is not a horizontal partition direction based on the height of the current block being greater than the first threshold.

4. The method of claim **1**, wherein the determining includes at least one of:

determining that the partition size of the SBT mode is not a half size of the current block based on (i) the first threshold being 8 and (ii) the at least one allowed block partitioning structure including a binary tree partitioning structure; and

determining that the partition size of the SBT mode is not a quarter size of the current block based on (i) the first threshold being 16 and (ii) the at least one allowed block partitioning structure including a triple tree partitioning structure.

5. The method of claim **1**, wherein the determining includes:

determining the partition direction and the partition size of the SBT mode for the current block based on (i) the current block being a chroma block and (ii) sizes of a plurality of sub-blocks of the current block being greater than a second threshold, the current block being partitioned into the plurality of sub-blocks based on the partition direction and the partition size of the SBT mode.

6. The method of claim **1**, wherein a flag indicating the partition direction of the SBT mode is not included in the prediction information.

7. The method of claim **1**, wherein a flag indicating the partition size of the SBT mode is not included in the prediction information.

8. The method of claim **1**, wherein a flag indicating whether the SBT mode is used for the current block is not included in the prediction information.

9. The method of claim **1**, further comprising:

determining a context model for a flag based on a partitioning depth associated with the current block, the flag indicating whether the SBT mode is used for the current block.

10. The method of claim **1**, further comprising:

determining a context model for a flag based on at least one of the partition direction of the SBT mode, a width of the current block, and a height of the current block, the flag indicating the partition size of the SBT mode.

11. The method of claim **1**, wherein the prediction information indicates that the current block is a non-merge inter block.

12. A video encoding apparatus, comprising:

processing circuitry configured to:

generate prediction information of a current block of a coding unit tree, the prediction information indicating that at least one block partitioning structure is allowed for the current block and that a sub-block transform (SBT) mode is used for the current block;

determine a partition direction and a partition size of the SBT mode for the current block based on the at least one allowed block partitioning structure indicated in the prediction information and based on a comparison between at least one dimension of the current block and a first threshold, a partition of the current block based on the partition direction and the partition size of the SBT mode being different from at least one partition of the current block based on the at least one allowed block partitioning structure indicated in the prediction information; and

encode the current block based on the determined partition direction and partition size of the SBT mode.

13. The apparatus of claim **12**, wherein the processing circuitry is further configured to:

determine the partition direction of the SBT mode for the current block based on (i) one of a width and a height of the current block being greater than the first threshold and (ii) a partitioning depth associated with the current block being less than a maximum allowed partitioning depth of the coding unit tree.

14. The apparatus of claim **13**, wherein the processing circuitry is further configured to perform at least one of:

determining that the partition direction of the SBT mode is not a vertical partition direction based on the width of the current block being greater than the first threshold; and

determining that the partition direction of the SBT mode is not a horizontal partition direction based on the height of the current block being greater than the first threshold.

15. The apparatus of claim **12**, wherein the processing circuitry is further configured to perform at least one of:

determining that the partition size of the SBT mode is not a half size of the current block based on (i) the first threshold being 8 and (ii) the at least one allowed block partitioning structure including a binary tree partitioning structure; and

determining that the partition size of the SBT mode is not a quarter size of the current block based on (i) the first threshold being 16 and (ii) the at least one allowed block partitioning structure including a triple tree partitioning structure.

16. The apparatus of claim **12**, wherein the processing circuitry is further configured to:

determine the partition direction and the partition size of the SBT mode for the current block based on (i) the current block being a chroma block and (ii) sizes of a plurality of sub-blocks of the current block being greater than a second threshold, the current block being partitioned into the plurality of sub-blocks based on the partition direction and the partition size of the SBT mode.

17. The apparatus of claim **12**, wherein a flag indicating the partition direction of the SBT mode is not included in the prediction information.

18. The apparatus of claim **12**, wherein a flag indicating the partition size of the SBT mode is not included in the prediction information.

19. The apparatus of claim **12**, wherein a flag indicating whether the SBT mode is used for the current block is not included in the prediction information.

20. A non-transitory computer-readable storage medium storing a program executable by at least one processor to perform:

generating prediction information of a current block of a coding unit tree, the prediction information indicating that at least one block partitioning structure is allowed for the current block and that a sub-block transform (SBT) mode is used for the current block;

determining a partition direction and a partition size of the SBT mode for the current block based on the at least one allowed block partitioning structure indicated in the prediction information and based on a comparison

between at least one dimension of the current block and a first threshold, a partition of the current block based on the partition direction and the partition size of the SBT mode being different from at least one partition of the current block based on the at least one allowed block partitioning structure indicated in the prediction information; and

encoding the current block based on the determined partition direction and partition size of the SBT mode.

* * * * *